Operating Systems and Languages Library

MS-DOS

User Guide

olivetti

PREFACE

This manual is a user guide for the MS-DOS operating system. It describes Microsoft Version 3.10 of MS-DOS, and is for anyone who wishes to use this operating system on an Olivetti Personal Computer.

SUMMARY

The first chapter provides a general introduction to MS-DOS.

Chapter 2 tells you how to start MS-DOS, and describes the steps you need to take to operate your system for the first time, such as formatting disks.

Chapter 3, 4 and 5 describe in more detail the major features of MS-DOS, such as its directory structure.

Chapter 6 to 10 provide detailed reference information on MS-DOS commands, the Video File Editor (EDIT), the Line Editor (EDLIN), the Linker (MS-LINK) and the Debugger (DEBUG). Command descriptions include the command purpose, syntax, characteristics and examples.

RELATED PUBLICATIONS:

Installation and Operations Guide for your Personal Computer

MS-DOS Fundamental Operations Guide

MS GW-BASIC Interpreter under MS-DOS User Guide (Code 4001420 H)

MS-DOS System Programmer Guide (Code 4024270 M)

MS-DOS Quick Reference Guide (Code 4024250 K)

DISTRIBUTION: General (G)

FIRST EDITION: February 1986

PUBLICATION ISSUED BY:

Ing. C. Olivetti & C., S.p.A.
Direzione Documentazione
77, Via Jervis - 10015 Ivrea (Italy)

- OLIVETTI is a trademark of Ing. C. Olivetti & C., S.p.A.
- OLITALK is a trademark of Ing. C. Olivetti & C., S.p.A.
- OLITERM is a trademark of Ing. C. Olivetti & C., S.p.A.
- SORTP is a trademark of Ing. C. Olivetti & C., S.p.A.
- ASM-86 is a trademark of Digital Research
- CB-86 is a trademark of Digital Research
- CBASIC-86 is a trademark of Digital Research
- CLEO is a trademark of Phone 1 Inc.
- Concurrent CP/M-86 is a trademark of Digital Research
- Concurrent DOS is a trademark of Digital Research
- CP/M-86 is a trademark of Digital Research
- DDT-86 is a trademark of Digital Research
- Dr. Draw is a trademark of Digital Research
- Dr. Graph is a trademark of Digital Research
- Dr. Logo is a trademark of Digital Research
- ETHERNET is a trademark of Xerox Corp.
- GEM COLLECTION is a trademark of Digital Research
- GEM DESKTOP is a trademark of Digital Research
- GEM DRAW is a trademark of Digital Research
- GEM GRAPH is a trademark of Digital Research
- GEM PAINT is a trademark of Digital Research
- GEM WORDCHART is a trademark of Digital Research
- GEM WRITE is a trademark of Digital Research
- GSS is a registered trademark of Graphic Software Systems, Inc.
- GSS-CHART is a trademark of Graphic Software Systems, Inc.
- GSS-DRIVERS is a trademark of Graphic Software Systems, Inc.
- GSS-KERNEL SYSTEM is a trademark of Graphic Software Systems, Inc.
- GSS-TOOLKIT is a trademark of Graphic Software Systems, Inc.
- GSX-86 is a trademark of Digital Research
- GW is a trademark of Microsoft Corp.
- IBM is a registered trademark of International Business Machines Corp.
- MICROSOFT is a registered trademark of Microsoft Corp.
- MS is a trademark of Microsoft Corp.
- OMNINET is a trademark of Corvus Systems Inc.
- p-System is a trademark of Softech Microsystem, Inc
- PC-DOS is a trademark of International Business Machines Corp.
- PEACHPAK is a trademark of Peachtree Software International Ltd.
- Personal Basic is a trademark of Digital Research
- SID-86 is a trademark of Digital Research
- TerminAll is a trademark of TOPIC System, Inc.
- TOPIC is a trademark of TOPIC System, Inc.
- UCSD and UCSD Pascal are registered trademarks of the Regent of the University of California
- UNIX is a trademark of Bell Laboratories
- Z80 is a registered trademark of Zilog Inc.
- Z8000 is a registered trademark of Zilog Inc.

CONTENTS

1. INTRODUCTION

WHAT IS MS-DOS? 1-1

COMMAND LIBRARY 1-1

FILE AND DIRECTORY HANDLING 1-1

PROGRAMMING TOOLS 1-1

INTERNAL AND EXTERNAL COMMANDS 1-2

BATCH PROCESSING 1-2

THE AUTOEXEC.BAT FILE 1-2

REAL-TIME CLOCK FUNCTIONS 1-2

INTERFACE HANDLING 1-3

DISKS 1-3

DISKETTE HANDLING 1-4

PURCHASING DISKETTES 1-4

LABELING DISKETTES 1-4

WRITE-PROTECTION 1-4

NOTATION CONVENTIONS 1-5

2. GETTING STARTED

HOW TO LOAD MS-DOS 2-1

HOW TO TERMINATE YOUR WORKING SESSION 2-1

THE DEFAULT DRIVE 2-2

DRIVE LETTERS FOR SINGLE DRIVE SYSTEMS 2-3

HOW TO FORMAT YOUR DISKETTES 2-3

HOW TO BACK-UP YOUR DISKETTES 2-5

COPYING DISKETTES USING ONLY ONE DRIVE 2-7

HOW TO LIST THE FILES ON A DISK 2-8

AUTOMATIC PROGRAM EXECUTION 2-9

SETTING UP YOUR DISK FOR NATIONAL SUPPORT 2-9

KEYBOARD AND SCREEN SUPPORT 2-9

MS-DOS COUNTRY SPECIFIC OPERATIONS SUPPORT 2-11

DISPLAYING THE CHARACTER SET 2-12

HOW TO PREPARE THE HARD DISK 2-12

HOW TO SET UP YOUR HARD DISK FOR MS-DOS ONLY 2-13

HOW TO PARTITION YOUR HARD DISK 2-18

HOW TO CREATE THE MS-DOS PARTITION 2-20

3. MS-DOS CONTROL KEYS AND EDITING FUNCTION KEYS

INTRODUCTION 3-1

MS-DOS CONTROL KEYS 3-1

EDITING FUNCTION KEYS 3-4

4. FILES AND DIRECTORIES

HOW MS-DOS KEEPS TRACK OF YOUR FILES 4-1

HOW TO NAME YOUR FILES 4-1

WILD CARDS 4-2

THE ? WILD CARD 4-2

THE * WILD CARD 4-3

RESERVED DEVICE NAMES 4-4

HOW TO COPY YOUR FILES 4-5

HOW TO PROTECT YOUR FILES 4-6

HOW TO EXAMINE THE CONTENTS OF A FILE 4-6

DIRECTORIES 4-6

FILE NAMES AND PATHS 4-9

PATHNAMES 4-9

PATHS AND EXTERNAL COMMANDS 4-11

HOW TO DISPLAY YOUR CURRENT DIRECTORY 4-12

HOW TO CREATE A DIRECTORY 4-14

HOW TO CHANGE YOUR CURRENT DIRECTORY 4-14

HOW TO REMOVE A DIRECTORY 4-15

HOW TO CHECK THE VALIDITY OF YOUR FILES 4-15

5. ENTERING AND USING MS-DOS COMMANDS

COMMAND SYNTAX 5-1

PARAMETERS 5-2

INTERNAL AND EXTERNAL COMMANDS 5-6

ENTERING AND USING MS-DOS COMMANDS 5-6

EXTERNAL COMMAND SYNTAX 5-8

INFORMATION COMMON TO ALL MS-DOS COMMANDS 5-9

BATCH PROCESSING 5-10

HOW TO CREATE AND EXECUTE A BATCH FILE 5-11

THE AUTOEXEC.BAT FILE 5-14

BATCH FILE WITH REPLACEABLE PARAMETERS 5-15

INPUT AND OUTPUT 5-16

REDIRECTING YOUR OUTPUT 5-16

REDIRECTING YOUR INPUT DIR >>file 5-17

FILTERS 5-17

COMMAND PIPING 5-18

6. COMMANDS

INTRODUCTION 6-1

MS-DOS 3.1 AND NETWORKING 6-5

ASSIGN 6-8

ATTRIB 6-10

BACKUP 6-11

BREAK 6-14

CHDIR 6-15

CHKDSK 6-17

CLS 6-20

COMMAND 6-21

COMP 6-24

COPY 6-27

CTTY 6-34

DATE	6-35
DEL	6-38
DIR	6-40
DISKCOMP	6-41
DISKCOPY	6-45
ECHO	6-47
ERASE	6-47
EXE2BIN	6-48
EXIT	6-50
FC	6-51
FDISK	6-60
FIND	6-61
FOR	6-64
FORMAT	6-67
GOTO	6-70
GRAPHICS	6-72
GRAFTABL	6-75
GWBASIC	6-76
HEXDUMP	6-77
IF	6-80
JOIN	6-82
LABEL	6-87

MKDIR 6-88

MODE 6-89

MODE COM: 6-90

MODE n 6-92

MODE LPT 6-94

MORE 6-96

PATH 6-97

PAUSE 6-99

PRINT 6-100

PROMPT 6-104

RECOVER 6-107

REM 6-109

REN 6-109

RESTORE 6-111

RMDIR 6-115

SELECT 6-116

SET 6-118

SHARE 6-120

SHIFT 6-122

SORT 6-123

SUBST 6-125

SYS 6-129

TIME 6-130

TREE 6-132

TYPE 6-133

VER 6-134

VERIFY 6-135

VOL 6-136

WRITECHK 6-137

7. VIDEO FILE EDITOR (EDIT)

INTRODUCTION 7-1

HOW TO INVOKE THE VIDEO FILE EDITOR 7-1

THE DISPLAY 7-3

THE KEYBOARD 7-4

EDIT 7-7

GENERAL EDITING FUNCTION KEYS 7-8

WINDOW MOVING FUNCTION KEYS 7-16

EXITING AND SAVING FUNCTION KEYS 7-18

COMMANDS AND SEARCHING 7-19

STRING SEARCHES 7-19

COMMANDS 7-21

GOTO 7-21

DELETE 7-22

FILE 7-22

8. **LINE EDITOR (EDLIN)**

INTRODUCTION **8-1**

HOW TO INVOKE THE EDLIN PROGRAM **8-2**

EDLIN 8-2

INTER-LINE COMMANDS **8-5**

line (EDIT LINE) **8-6**

A (APPEND LINES) **8-8**

C (COPY LINES) **8-9**

D (DELETE LINES) **8-12**

E (END EDITING) **8-15**

I (INSERT LINES) **8-16**

L (LIST TEXT) **8-20**

M (MOVE LINES) **8-23**

P (PAGE) **8-25**

Q (QUIT EDITING) **8-26**

R (REPLACE TEXT) **8-27**

S (SEARCH TEXT) **8-30**

T (TRANSFER LINES) **8-34**

W (WRITE LINES) **8-35**

INTRA-LINE COMMANDS **8-36**

COPY1 **8-37**

COPYTO **8-38**

COPYLINE **8-40**

SKIP1 8-41

SKIPTO 8-42

KILL 8-44

INS 8-46

NEWTEMP 8-47

9. THE LINKER

INTRODUCTION 9-1

TEMPORARY FILES 9-1

CHANGING DISKETTES 9-2

SEGMENTS, GROUPS AND CLASSES 9-2

INTERACTIVE ENTRY 9-3

COMMAND LINE ENTRY 9-10

AUTOMATIC RESPONSE FILE ENTRY 9-13

10.THE DEBUGGER

INTRODUCTION 10-1

HOW TO INVOKE THE DEBUG PROGRAM 10-1

DEBUG 10-1

DEBUGGING COMMANDS 10-4

COMMAND PARAMETERS 10-4

A (ASSEMBLE) 10-7

C (COMPARE) 10-10

D (DUMP) 10-11

E (ENTER) 10-14

F (FILL) 10-17

G (GO) 10-18

H (HEX) 10-21

I (INPUT) 10-23

L (LOAD) 10-24

M (MOVE) 10-27

N (NAME) 10-29

O (OUTPUT) 10-31

P (PROCEED) 10-32

Q (QUIT) 10-33

R (REGISTER) 10-34

S (SEARCH) 10-37

T (TRACE) 10-39

U (UNASSEMBLE) 10-40

W (WRITE) 10-42

A. ASCII DISPLAY AND KEYBOARD CODE TABLES

BASIC ASCII CODE (ISO 7-bit code) A-1

EXTENDED ASCII CODE FOR THE DISPLAY (ISO 8-bit code) A-5

NATIONAL VARIATIONS IN EXTENDED ASCII CODE (ISO 8-bit code) A-8

EXTENDED KEYBOARD A-14

B. ANSI ESCAPE SEQUENCES

INTRODUCTION B-1

CURSOR MOVING FUNCTIONS B-2

ERASING FUNCTIONS B-4

GRAPHIC MODE FUNCTIONS B-5

KEY REASSIGNMENT B-7

C. CONFIGURING MS-DOS

INTRODUCTION C-1

CONFIGURING COMMANDS C-1

BREAK C-1

BUFFERS C-2

COUNTRY C-3

DEVICE C-5

FCBS C-5

FILES C-6

LASTDRIVE C-7

SHELL C-8

D. ERROR MESSAGES D-1

E. GLOSSARY OF TERMS E-1

F. THE VIRTUAL DISK SYSTEM F-1

INDEX

1. INTRODUCTION

ABOUT THIS CHAPTER

This chapter introduces some of the more commonly used features of MS-DOS, provides some information on disk handling and defines the notation conventions used throughout this book.

CONTENTS

WHAT IS MS-DOS? 1-1

COMMAND LIBRARY 1-1

FILE AND DIRECTORY HANDLING 1-1

PROGRAMMING TOOLS 1-1

INTERNAL AND EXTERNAL COMMANDS 1-2

BATCH PROCESSING 1-2

THE AUTOEXEC.BAT FILE 1-2

REAL-TIME CLOCK FUNCTIONS 1-2

INTERFACE HANDLING 1-3

DISKS 1-3

DISKETTE HANDLING 1-4

PURCHASING DISKETTES 1-4

LABELING DISKETTES 1-4

WRITE-PROTECTION 1-4

NOTATION CONVENTIONS 1-5

WHAT IS MS-DOS?

The Microsoft Disk Operating System (MS-DOS) is a group of programs that controls the running and operation of your computer. It provides an interface between you and your computer.

Through MS-DOS you communicate with the Central Processing Unit (CPU), monitor, disk drives, printer, and other peripherals. The Disk Operating System enables you to manipulate program and data files stored on diskette or hard disk.

MS-DOS is supplied on your MS-DOS diskette.

The major features of MS-DOS are outlined in the sections that follow.

COMMAND LIBRARY

MS-DOS has a command library of over 40 commands that provide you with an environment suitable for handling files of information, developing programs, and running applications.

FILE AND DIRECTORY HANDLING

File handling commands not only allow you to copy and delete files, copy entire diskettes, display the contents of files, etc., but also to group files into directories at your convenience. Moreover, MS-DOS enables you to create directories within a directory, thereby creating a hierarchical directory structure. Refer to Chapter 4 for details.

PROGRAMMING TOOLS

A set of programming tools enable you to write and develop programs. You can edit program files using the Video File Editor (see Chapter 7) or Line Editor EDLIN (see Chapter 8), link object files using the LINK utility (see Chapter 9), and debug executable files using the DEBUG utility (see Chapter 10). Macro Assembler, PASCAL, FORTRAN and other high level languages are separately available, to produce executable files. The GWBASIC interpreter is available on your system disk for interpretive programming.

INTERNAL AND EXTERNAL COMMANDS

When MS-DOS is initialized some commands are loaded into memory and reside there. Other commands remain on disk. The former are known as internal commands, the latter are external commands and are loaded into memory and executed when required. With most of these external commands, after they are executed, they are removed from memory, thereby optimizing the use of memory. However a few of these external commands, (GRAFTABL, GRAPHICS, PRINT, and SHARE), remain resident in memory after they have executed (see Chapter 6 for details of these commands).

BATCH PROCESSING

MS-DOS enables a commonly executed series of commands to be grouped into one file - a batch file - that can be executed simply by entering the file name. Refer to Chapter 5 for details.

THE AUTOEXEC.BAT FILE

The AUTOEXEC.BAT file is a special batch file which, if present, is executed automatically at system initialization. This is useful if your application requires a certain sequence of commands to be executed every time the system is initialized (see Chapter 5).

REAL-TIME CLOCK FUNCTIONS

When no AUTOEXEC.BAT file is present MS-DOS asks you the DATE and TIME.

MS-DOS has two commands that utilize the Real-Time Clock. These are:

- DATE which enables you to set the date

- TIME which enables you to set the time

These are important not only for programs that use time-dependent functions, but also because MS-DOS provides you with information about the time and date of creation or modification of your files.

INTERFACE HANDLING

MS-DOS allows you to communicate with compatible devices (plotters, printers modems, etc.) via the standard RS-232-C serial interface. You will need to set the protocol for the interface using the MODE command (see Chapter 6).

DISKS

Information is stored either on 5 1/4 in. floppy disks or, if your system has one, on hard disk. This manual will refer to the former as diskettes and the latter as the hard disk. The term ''disk'' will be used to mean either diskette or hard disk.

Drive letters (A, B and C) are the means by which commands can identify a particular drive.

The drive letter of the first diskette drive in any system is A. The drive letter of any second diskette drive is B. The drive letter of any hard disk is C.

The drive capacity governs the type of diskette that can be used in that drive.

Diskettes can have a variety of capacities to hold data; as illustrated in the following table. See your Installation and Operations Guide to check what Diskette capacities your Disk Drive(s) can read and write.

5 1/4 inch Diskette Capacities

	Double Density **48 t.p.i.**		**High Density** **96 t.p.i.**		
	8 sector	9 sector	8 sector	9 sector	15 sector
Single Sided	160 KB	180 KB	-	-	-
Double Sided	320 KB	360 KB	640 KB*	720 KB*	1.2 MB

* Not supported by MSDOS ver 3.10

DISKETTE HANDLING

Although diskettes are generally durable, damage to diskettes will be minimized if you take the following precautions:

- Never bend diskettes
- Do not touch the exposed surface of the diskette
- Always keep the diskette in its cardboard envelope when not in use and store it in the diskette carton
- Keep dust out of the diskette drives by keeping the drive covers closed when not in use.

PURCHASING DISKETTES

When purchasing your media make sure that the diskettes are Double Sided for Double Sided Disk Drives, and the diskettes are 96 tracks per inch for High Density disk drives.

LABELING DISKETTES

Every carton of diskettes contains a supply of self-adhesive labels for identifying diskettes. It is good practice to write all relevant details on the label before attaching it to the diskette. But if you do find it necessary to write on the label after sticking it to the diskette, you should avoid using sharp pencils or ball-point pens as these may damage the surface of the diskette. In this case a felt-tipped pen is recommended.

WRITE-PROTECTION

To protect your data from being accidentally overwritten, you can apply write-protection to your diskettes. A sheet of aluminized write-protect labels is provided with every carton of diskettes. To apply write-protection, simply fix an aluminized label over the write-protect notch cut into the side of the diskette. To remove write-protection, simply remove the label.

NOTATION CONVENTIONS

The following notation conventions are used throughout this book:

- Uppercase, bold letters and words within a syntax line represent keywords that must be typed exactly as shown.

  For example in the command line:

  DISKCOPY [*sourcedrive*:] [*targetdrive*:]

  DISKCOPY should be typed as shown.

  Outside syntax lines, keywords are shown in uppercase but not in bold.

  Note that uppercase letters and words are used simply as visual aids in this manual. Keywords may be typed in lowercase if desired.

- Lowercase italicized characters and words represent parameter names. They indicate that variable information is to be provided by the user.

  For example in the command line:

  DISKCOPY A: B:

  both *sourcedrive* and *targetdrive* have been replaced by specific values, that is **A** and **B**.

- Hyphens may join lowercase letters or words to form a single parameter name.

For example in the command line:

R [*register-name* | **F**]

register-name is a single variable item that should be replaced by a single specific value, for example AX.

- A blank, a comma, a colon, or a semicolon may be used to separate the items in a line. In this manual the blank is usually shown in syntax lines.

- The symbols listed below are used to define the syntax of a line, but should not be typed in the actual line:

  [ ] brackets

  { } braces

  | vertical stroke (''or'' sign)

  ... ellipsis

- Items contained by brackets ([ ]) are optional and so may or may not be selected.

  For example the representation:

  [*filespec*]

  indicates that a *filespec* may or may not be entered.

- Items enclosed by braces ({ }) and separated by vertical strokes (|) are alternatives. You should select only one such alternative.

  For example the representation:

  {A|B|C}

  indicates that either A or B or C should be selected.

- Items enclosed by brackets ([ ]) and separated by vertical strokes (|) are optional alternatives. You may choose one such alternative, or none at all.

For example the representation:

[A|B|C]

indicates that A or B or C may, but need not, be selected.

- An ellipsis indicates that the preceding item or group of items may be repeated more than once in succession.

  For example the representation:

  A [B]...

  indicates that A can be typed alone or can be followed by

  B

  more than once in succession.

- Letters and words in bold indicate MS-DOS messages that appear on your Personal Computer screen. For example:

  Insert new diskette for drive B: and strike ENTER when ready

- Letters and words shown in condensed bold indicate that you must press a specific key. For example the key whose inscription is CTRL is always referred to as **CTRL**.

- Commands need to be confirmed by typing ↵ (the **ENTER** key), at the end of the command line.

- by performing a system reset by pressing the **CTRL**, **ALT**, and **DEL** keys simultaneously, or by pressing the hardware reset button located on the front panel beneath the diskette drives. If you have a hard disk, do **not** reset when the hard disk is in use, as you may corrupt your files.

THE DEFAULT DRIVE

The **A**> (or **C**>) is the MS-DOS prompt from the command processor. It tells you that MS-DOS is ready to accept commands. For systems that do not have a hard disk you will always load MS-DOS from drive A.

The **A** (or **C**) in the MS-DOS prompt indicates the default disk drive. This means that MS-DOS will search only that drive for any commands or other file names that you enter and will perform the specified task on that disk unless you specify a different drive.

For example, if following the **A**> prompt you type:

DIR
then press the **ENTER** key

then the DIR command is executed on the default drive. But if you type:

DIR B:
then press the **ENTER** key

the DIR command is executed on drive B, but drive A remains the default drive. To change the default drive to drive B you must type:

B:
then press the **ENTER** key.

and MS-DOS will respond:

B>

Subsequently, MS-DOS will search only the diskette in drive B until you specify a different default drive.

DRIVE LETTERS FOR SINGLE DRIVE SYSTEMS

If you have a single diskette drive system it is as though you had a system with two diskette drives, except that drive **A** and drive **B** represent two diskettes instead of two drives. You enter commands exactly as you would using a dual-drive system and MS-DOS will prompt you when you need to change diskettes.

HOW TO FORMAT YOUR DISKETTES

You must format all new diskettes before they can be used by MS-DOS.

A blank diskette must be formatted with the FORMAT command. The FORMAT command changes the format of the diskette to one that MS-DOS can use. If the diskette is not already blank, formatting it will destroy any data that exists on the diskette.

Formatting places the tracks on a diskette and creates an empty directory for that diskette. The directory is the means by which MS-DOS keeps track of what is on the diskette.

If you type:

```
FORMAT B:
```

then press the **ENTER** key.

FORMAT issues the following message:

```
Insert new diskette for drive B:
and strike ENTER when ready
```

After you insert the new diskette in drive B and press any key on the keyboard, formatting begins and the system keeps you informed of the progress of the FORMAT command by responding:

```
Formatting ... n
```

Where *n* is the number of the track being formatted.

When formatting is complete, you will see on your screen a message similar to this:

```
        362496 bytes total disk space
        362496 bytes available on disk

Format another (Y/N)?
```

The DISKCOPY command copies the contents of one diskette onto another. You can use this command to duplicate the MS-DOS system diskette, or a diskette that contains your own files. DISKCOPY is the fastest way of copying a diskette because it copies the entire diskette in one operation, including MS-DOS system files if they exist. However, note that the DISKCOPY command can only be used where the source and target diskettes are the same capacity.

For example, you will need to make a copy of your MS-DOS system diskette. Insert it in drive A and type:

```
DISKCOPY A: B:
```
then press **ENTER**

DISKCOPY responds:

```
Insert source diskette in drive A:
Insert destination diskette in drive B:
Strike any key when ready
```

Make sure the MS-DOS system diskette is in drive A and insert a blank diskette in drive B. Press any character key after you have done this and MS-DOS will begin copying the DISKCOPY system diskette.

The following message appears on the screen:

```
Copying 2 side(s), 9 sectors per track
```

If your target diskette was not formatted, DISKCOPY will format it as it copies. After MS-DOS has copied the diskette, MS-DOS displays:

```
Copy complete
Copy another disk (Y/N)?
```

Press **Y** (for Yes) if you wish to copy another diskette with the DISKCOPY command. If you press **N** (for No), the default drive prompt is displayed.

Note

If either of the diskettes that you are using has defective tracks, the DISKCOPY command will not work reliably. Use the COPY command to back-up your diskettes in these cases. COPY will skip over defective tracks. But note that some of the system files are ''hidden'' and cannot be copied using the COPY command. Therefore, before copying your system diskette using the COPY command you must format the target diskette using the FORMAT command with the **/S** option.

COPYING DISKETTES USING ONLY ONE DRIVE

If you have a a single-drive system or a hard disk system with one diskette drive, you must first insert the system diskette into drive A and type:

DISKCOPY
then press **ENTER**

DISKCOPY then replies with the message

```
Insert source diskette in drive A:
Strike any key when ready
```

After striking any key DISKCOPY responds

```
Copying 2 side(s), 9 sectors per track
```

DISKCOPY begins copying the contents of the diskette to memory. When available memory space is full, MS-DOS prompts:

```
Insert destination diskette in drive A:
Strike any key when ready
```

On doing so, DISKCOPY transfers the data copied to memory onto the target diskette. You are then prompted to reinsert the source diskette.

After performing this sequence a number of times, the entire source diskette will have been copied to the target diskette. DISKCOPY then displays:

```
Copy complete
Copy another disk (Y/N)?
```

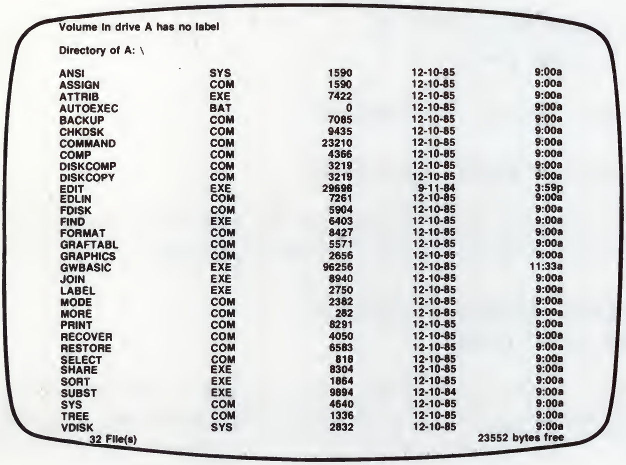
CAUTION: You must make sure that you insert the correct source and target diskettes as DISKCOPY does not know the difference. Moreover, you are strongly recommended to write-protect your source diskette to avoid inadvertently overwriting it.

HOW TO LIST THE FILES ON A DISK

When a disk is formatted a directory is placed on that disk. This directory provides the means by which MS-DOS keeps track of the files on that disk. The DIR command enables you to display the contents of this directory and hence obtain a list of the files held on that disk. For example, if your copy of the MS-DOS system diskette is in drive A and you type:

DIR A:
then press **ENTER**

MS-DOS will respond with a directory listing of all the files in the current directory on your MS-DOS system diskette. The display will look similar to this:



```
Volume in drive A has no label

Directory of A: \

ANSI        SYS      1590   12-10-85    9:00a
ASSIGN      COM      1590   12-10-85    9:00a
ATTRIB      EXE      7422   12-10-85    9:00a
AUTOEXEC    BAT         0   12-10-85    9:00a
BACKUP      COM      7085   12-10-85    9:00a
CHKDSK      COM      9435   12-10-85    9:00a
COMMAND     COM     23210   12-10-85    9:00a
COMP        COM      4366   12-10-85    9:00a
DISKCOMP    COM      3219   12-10-85    9:00a
DISKCOPY    COM      3219   12-10-85    9:00a
EDIT        EXE     29698    9-11-84    3:59p
EDLIN       COM      7261   12-10-85    9:00a
FDISK       COM      5904   12-10-85    9:00a
FIND        EXE      6403   12-10-85    9:00a
FORMAT      COM      8427   12-10-85    9:00a
GRAFTABL    COM      5571   12-10-85    9:00a
GRAPHICS    COM      2656   12-10-85    9:00a
GWBASIC     EXE     96256   12-10-85   11:33a
JOIN        EXE      8940   12-10-85    9:00a
LABEL       EXE      2750   12-10-85    9:00a
MODE        COM      2382   12-10-85    9:00a
MORE        COM       282   12-10-85    9:00a
PRINT       COM      8291   12-10-85    9:00a
RECOVER     COM      4050   12-10-85    9:00a
RESTORE     COM      6583   12-10-85    9:00a
SELECT      COM       818   12-10-85    9:00a
SHARE       EXE      8304   12-10-85    9:00a
SORT        EXE      1864   12-10-85    9:00a
SUBST       EXE      9894   12-10-84    9:00a
SYS         COM      4640   12-10-85    9:00a
TREE        COM      1336   12-10-85    9:00a
VDISK       SYS      2832   12-10-85    9:00a
     32 File(s)                  23552 bytes free
```

Fig. 2-2 Directory Example

Two MS-DOS system files, IO.SYS and MSDOS.SYS are "hidden" files and will not appear when you issue the DIR command.

For each entry in the directory the size of the file in bytes is given, along with the date and time at which the file was created or last modified.

You can also get information about any file on your disk by entering DIR and a file name. For example, if you have created a file named MYFILE.TXT on the disk in the default drive, and you type:

```
DIR MYFILE.TXT
```

then press **ENTER**

MS-DOS will respond with a display of all the directory information for the file MYFILE.TXT.

AUTOMATIC PROGRAM EXECUTION

If you want to run a specific program or list of programs automatically each time you start MS-DOS, you can do so with automatic program execution. For example, you may want to have MS-DOS display the names of your files each time you load MS-DOS.

When you start MS-DOS, the command processor searches for a file named AUTOEXEC.BAT on the MS-DOS system diskette. This file is a program that MS-DOS will run each time MS-DOS is started. Chapter 5 tells you how to create an AUTOEXEC.BAT file.

SETTING UP YOUR DISK FOR NATIONAL SUPPORT

KEYBOARD AND SCREEN SUPPORT

When you start MS-DOS, input from the keyboard and output to the screen is handled by the BIOS (the Basic Input Output System). Initially this expects the keyboard to have USA characters on it. However your national keyboard may differ from the USA one, and the characters that appear on the screen may also differ. Your operating system disk needs to be set up so that it automatically reconfigures the BIOS to cope with your national requirements. See the Installation and Operations Guide for your computer, for details of the programs which reconfigure the BIOS.

If you have a USA ASCII keyboard your system will automatically interpret the characters you type at the keyboard correctly. If you have a national version, however, you must install an appropriate ''keyboard driver'' each time you start up your system and possibly you need to install an extension to the built-in ''font table''. A keyboard driver provides the tables that tell the system what code to generate for each key pressed. A font table provides the characters that are displayed on the screen. The command GRAFTABL contains the font tables for extensions to the ASCII character set (see Chapter 6 for details of this command and see Appendix A for the tables illustrating the font). Your particular national keyboard may require a different command to load your national font (see your computer's Installation and Operations Guide.

In addition to national version tables, your system contains tables for the standard USA ASCII keyboards. As soon as you reboot your system, the appropriate USA ASCII driver for your keyboard is reinstalled.

If you wish a specific keyboard driver and font to be loaded automatically immediately after bootstrap you must create an AUTOEXEC.BAT file with the national keyboard driver file name and font name as an instruction. Of course you have to ensure that the national keyboard driver is present on the system diskette you use to boot up your system. See Chapter 5 for more on AUTOEXEC.BAT files.

Note that it is not necessary to enter the file extension ''.COM'' when calling a national keyboard driver.

Note also that your system diskette may already be customized, with the appropriate AUTOEXEC.BAT file and transferred keyboard driver, when you receive it. If you are in doubt, try those keys on the keyboard that differ between USA ASCII and your national version. You will soon see whether you have the correct driver installed.

Calling The USA ASCII Driver

When a national keyboard driver is installed, it is possible to call the USA ASCII keyboard version by using the three-key combination **CTRL ALT F1** . When you do this, you can return to the national keyboard version by using another three-key combination, **CTRL ALT F2**.

Such swapping between national and USA ASCII versions may be necessary in order to use certain application or communication packages.

Key combinations are summarized as follows:

KEY COMBINATION	ACTION
CTRL ALT F1	Calls the USA ASCII keyboard driver.
CTRL ALT F2	Calls the national keyboard driver.

MS-DOS COUNTRY SPECIFIC OPERATIONS SUPPORT

DOS can be configured for different country's requirements for

- date
- time
- currency symbol
- decimal separator

This is achieved by setting the variable

COUNTRY = *country-code*

in the CONFIG.SYS file. (See Appendix C for more details).

DISPLAYING THE CHARACTER SET

To display the character corresponding to a given ASCII code (decimal 1-255), type:

ALT *number*

where *number* is the three-digit decimal code for the character. Use the numeric keypad **not** the numbers on the top row of the main body of the keyboard.

Note that some of the characters in the range 1-32 cannot be reproduced on the screen, using this technique. You cannot generate the ESC character (decimal 27) using this technique, use the Video Editor instead (see Chapter 7).

Appendix A shows the complete character set, for extended ASCII.

HOW TO PREPARE THE HARD DISK

If your Olivetti Personal Computer has a hard disk there are several steps you will need to take before you can use it with MS-DOS.

If MS-DOS will be the only operating system you will use, then you will need to assign the entire hard disk to MS-DOS as described in the section ''HOW TO SET UP YOUR HARD DISK FOR MS-DOS ONLY''. Note that the maximum hard disk size supported by MSDOS is 32 MB. If you have a hard disk with a capacity greater than 32 MB, see the section ''HOW TO PARTITION YOUR HARD DISK''.

If you intend to use operating systems in addition to MS-DOS, such as Concurrent DOS, UCSD p-System, or PCOS, then you need to divide your hard disk into separate areas called partitions. You will require one partition for each operating system. Each such partition must be set up using the operating system that will use that partition. The operations relevant to MS-DOS are described in the section entitled ''HOW TO PARTITION YOUR HARD DISK''. For information on how to set up the non-MS-DOS partitions refer to the appropriate operating system ''User Guide''.

After setting up your hard disk for MS-DOS only, you may subsequently wish to use other operating systems. This means that you will have to partition the hard disk. When you partition you destroy the information held on the hard disk. You therefore need to back-up the MS-DOS files that are on the hard disk using the BACKUP command, partition your hard disk for all the operating systems you require, format the MS-DOS partition, then restore the MS-DOS partition using the RESTORE command.

The MS-DOS command that you will use to set up your hard disk is called the FDISK command. It uses a series of video displays and prompts that guide you through the operations step by step. The example is based upon a 10 MB hard disk. Different size hard disks will result in different sizes shown on the display.

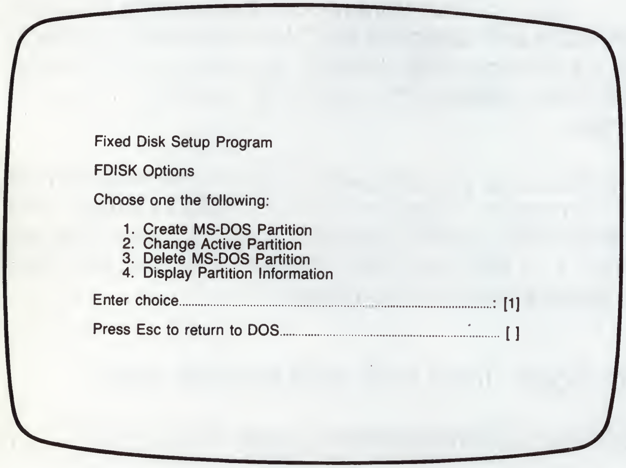
HOW TO SET UP YOUR HARD DISK FOR MS-DOS ONLY

1. Boot MS-DOS from system diskette in drive A and type:

   ```
   FDISK
   ```

   then press **ENTER**

The FDISK Options menu will then appear on the video display and will be similar to the following:



```
Fixed Disk Setup Program

FDISK Options

Choose one the following:

     1. Create MS-DOS Partition
     2. Change Active Partition
     3. Delete MS-DOS Partition
     4. Display Partition Information

Enter choice..........................................: [1]

Press Esc to return to DOS............................ [ ]
```

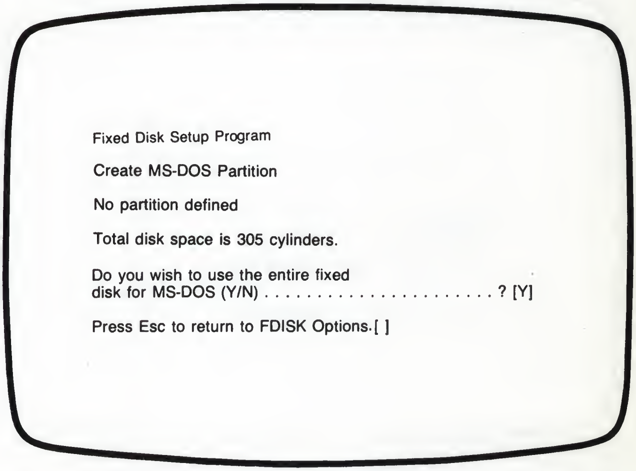
Fig. 2-3 FDISK Options Menu

2. Type:

   1

   then press **ENTER**

Assuming your hard disk has not already been set up, a video display similar to the following will appear:



```
Fixed Disk Setup Program

Create MS-DOS Partition

No partition defined

Total disk space is 305 cylinders.

Do you wish to use the entire fixed
disk for MS-DOS (Y/N) . . . . . . . . . . . . . . . . . . . . . . ? [Y]

Press Esc to return to FDISK Options.[ ]
```

Fig. 2-4 Create MS-DOS Partition Prompt

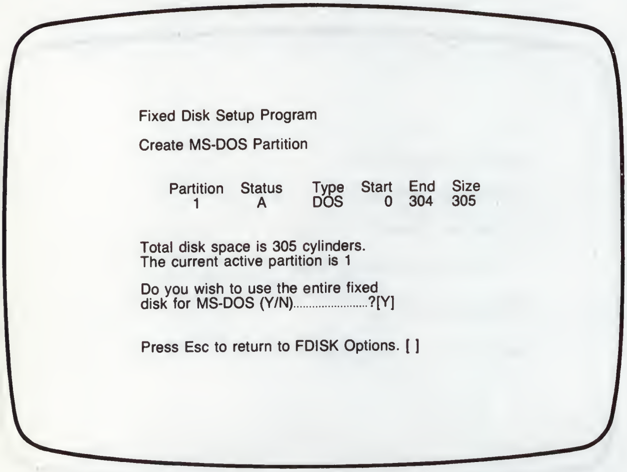
If your hard disk has already been set up, a different video display will appear. In this case refer to the section entitled ''HOW TO PARTITION YOUR HARD DISK''.

3. Type:

   Y

   then press **ENTER**

and the entire hard disk will be assigned to MS-DOS. A video display similar to the following will appear:



```
Fixed Disk Setup Program

Create MS-DOS Partition

     Partition  Status   Type  Start  End  Size
         1        A      DOS      0   304   305

Total disk space is 305 cylinders.
The current active partition is 1

Do you wish to use the entire fixed
disk for MS-DOS (Y/N)........................?[Y]

Press Esc to return to FDISK Options. [ ]
```

Fig. 2-5 MS-DOS Partition Created

4. Press **ESC**.

5. You should now have the display in Fig. 2-3. Press **ESC** to return to MS-DOS.

FORMATTING YOUR HARD DISK

1. You now need to format the hard disk so that MS-DOS can use it. Make sure that you have your system disk in drive A and A> is your MS-DOS prompt. Type:

   FORMAT C:/S/V
   then press **ENTER**

   This not only formats the hard disk but also copies the hidden files to it.

   If you do not want to boot MS-DOS from hard disk it is sufficient to type:

   FORMAT C:/V
   then press **ENTER**

In either case the following message appears:

```
WARNING, ALL DATA ON NON-REMOVABLE DISK
DRIVE C: WILL BE LOST!
Proceed with Format (Y/N)?
```

2. Type Y then press **ENTER**

```
Formatting... n
```

appears. The FORMAT program displays the track number it is formatting. After several minutes you will see the message:

```
Format complete
```

and if you entered **/S** in the command line the following message will also appear:

```
System transferred
```

This indicates that a copy of MS-DOS has been made on the hard disk. You will now be able to boot from the hard disk.

3. The following message then appears:

```
Volume label (11 characters, ENTER for none) ?
```

It is recommended that you give your hard disk a Volume label, for example type:

HARD DISK then press **ENTER**

You will get a message similar to the following:

```
10592256 bytes total disk space
61440 bytes used by system
10530816 bytes available on disk
```

4. If you have copied MS-DOS to the hard disk you will probably require the MS-DOS external commands to be copied there as well. With your MS-DOS system diskette still in drive A type:

```
COPY *.* C:/V
```

then press **ENTER**

This will copy all the non-hidden files from the system diskette to the hard disk, verifying the correctness of the copied files.

To test boot-strapping your system from the Hard Disk (provided you used the /S switch in formatting the hard disk).

1. Remove your diskette from Drive A.

2. Press **CTRL ALT DEL**

   If it fails start again with FDISK.

HOW TO PARTITION YOUR HARD DISK

Each operating system that supports the hard disk has a command that enables you to set up the partition that the particular operating system will use.

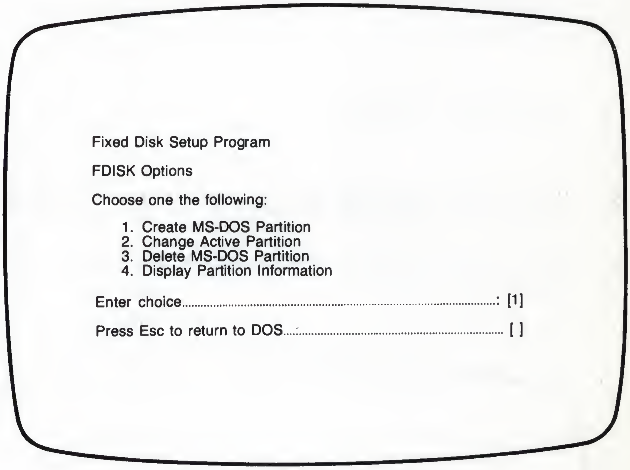
The FDISK command enables you to:

- Set up the MS-DOS partition to any size (up to 32MB and within the capacity of the hard disk) and at any location on the hard disk
- Change the partition that will be active when the system is rebooted
- Delete the MS-DOS partition
- Display the hard disk partition map

To invoke the FDISK program type:

FDISK
then press **ENTER**

The FDISK Options menu will be displayed as follows:



```
Fixed Disk Setup Program

FDISK Options

Choose one the following:

     1. Create MS-DOS Partition
     2. Change Active Partition
     3. Delete MS-DOS Partition
     4. Display Partition Information

Enter choice...............................................: [1]

Press Esc to return to DOS.................................. [ ]
```

Fig. 2-6 FDISK Options Menu

Proceed by entering the number corresponding to the selection you require.

HOW TO CREATE THE MS-DOS PARTITION

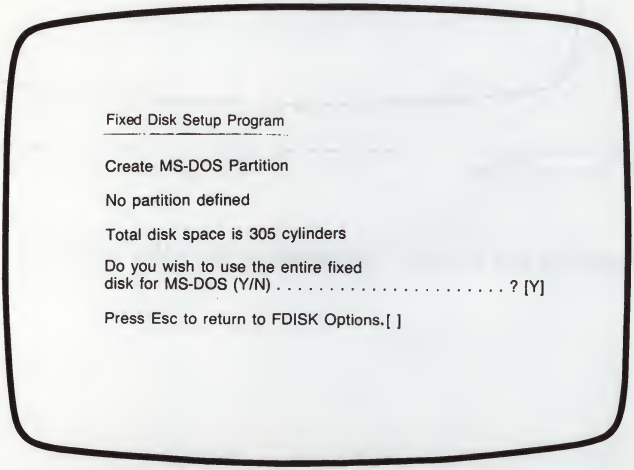
1. Choose selection 1 from the FDISK Options menu by typing:

   1

   then press **ENTER**

2. If the hard disk has already been set up, proceed to step 5.

3. If the hard disk has not already been set up, the following display appears:



```
Fixed Disk Setup Program

Create MS-DOS Partition

No partition defined

Total disk space is 305 cylinders

Do you wish to use the entire fixed
disk for MS-DOS (Y/N) . . . . . . . . . . . . . . . . . . . . . . ? [Y]

Press Esc to return to FDISK Options.[ ]
```

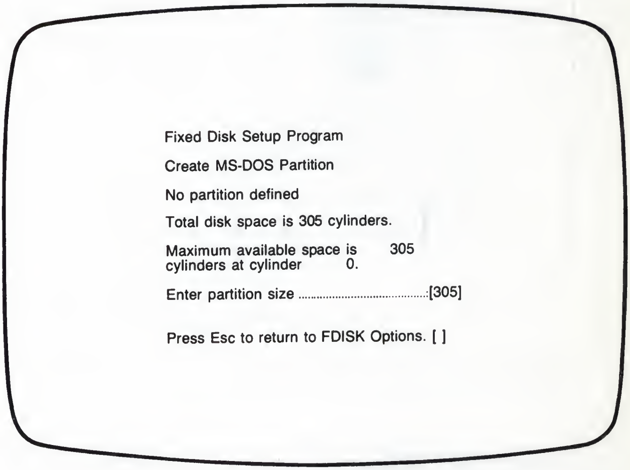
Fig. 2-7 Create MS-DOS Partition Prompt

4. Type

   N

   then press **ENTER**

   and a message similar to the following appears:

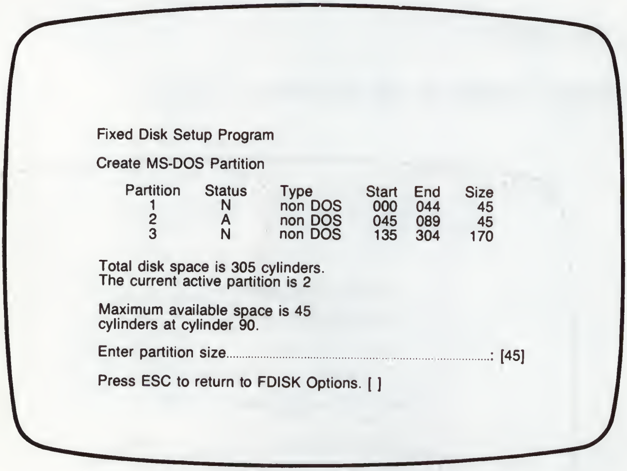


```
Fixed Disk Setup Program

Create MS-DOS Partition

No partition defined

Total disk space is 305 cylinders.

Maximum available space is      305
cylinders at cylinder        0.

Enter partition size ..........................................:[305]

Press Esc to return to FDISK Options. [ ]
```

Fig. 2-8 Create MS-DOS Partition Display

This display shows the total number of cylinders that your hard disk has. It also indicates the size and location of the largest number of contiguous cylinders available on the hard disk. As there is nothing on the hard disk, the number of contiguous cylinders available will be the same as the total hard disk space. Proceed to step 6.

5. If your hard disk has already been set up, a video display similar to the following will appear:



```
Fixed Disk Setup Program

Create MS-DOS Partition

     Partition   Status    Type       Start  End    Size
         1         N       non DOS     000   044     45
         2         A       non DOS     045   089     45
         3         N       non DOS     135   304    170

Total disk space is 305 cylinders.
The current active partition is 2

Maximum available space is 45
cylinders at cylinder 90.

Enter partition size.........................................: [45]

Press ESC to return to FDISK Options. [ ]
```

Fig. 2-9 Partition Status Display

The video display shown above illustrates a typical partition map of a hard disk having three partitions.

The display shows for each partition:

- The status of the partition; that is, whether or not the operating system in that partition will be booted when the system is rebooted from the hard disk. If so, it is designated the active partition (indicated by ''A''), while the other partitions are not active (indicated by ''N'').

- Whether or not the partition is type DOS or non DOS.

- The beginning and end of the partition in terms of the first and last cylinder numbers.

- The size of the partition.

- The capacity of the fixed disk, and the largest available space for another partition and the cylinder number at which this space starts.

6. FDISK prompts you as follows:

   Enter partition size.............:[xxxx]

   The default value is the maximum available space. Type either **ENTER** alone to accept the default value, or type the number of cylinders you require, followed by **ENTER**.

   You will then be prompted:

   Enter starting cylinder number..:[xxxx]

7. The default value for the starting cylinder number depends on the partition size that you have just specified. It is the first cylinder of the smallest space on the hard disk that is large enough for the partition. If this is where you want the MS-DOS partition to be, press **ENTER**. Otherwise you must enter the number of the cylinder that you wish to be the first MS-DOS cylinder, and then press **ENTER.** Your MS-DOS partition is now created.

8. Press **ESC** to return to the main FDISK options menu.

9. You now need to make the MS-DOS partition active.

How to Change the Active Partition

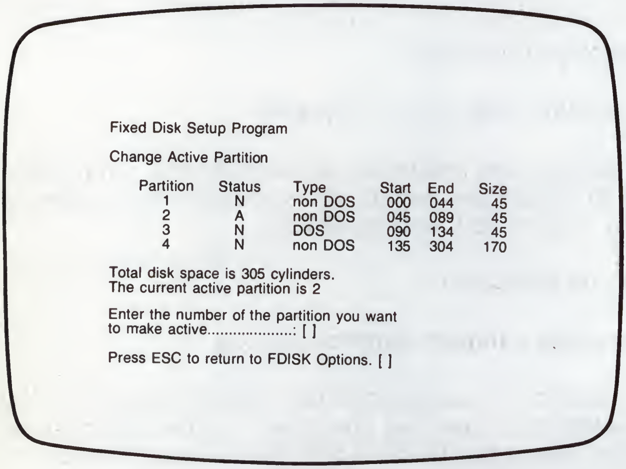
The active partition contains the operating system that will be active whenever you boot strap the system from the hard disk.

1. Choose selection 2 from the FDISK Options menu by typing:

   2

   then press **ENTER**

A video display similar to the following will appear:



```
Fixed Disk Setup Program

Change Active Partition

    Partition   Status     Type        Start   End    Size
        1          N       non DOS      000    044      45
        2          A       non DOS      045    089      45
        3          N       DOS          090    134      45
        4          N       non DOS      135    304     170

Total disk space is 305 cylinders.
The current active partition is 2

Enter the number of the partition you want
to make active....................: [ ]

Press ESC to return to FDISK Options. [ ]
```

Fig. 2-10 Change Active Partition Prompt

2. Enter the number of the partition whose operating system you wish to make active. For example:

    3

    then press **ENTER**

    makes the operating system in partition 3 active. The operating system in partition 2 then becomes non-active.

3. Press **ESC** to return to the FDISK menu, then **ESC** again to return MS-DOS.

4. You now need to format the active partion of the hard disk. Follow the instructions in the section ''FORMATTING YOUR HARD DISK'' above.

How to Delete the MS-DOS Partition

CAUTION: This option destroys the contents of the MS-DOS partition. Be sure to make a back-up of the MS-DOS partition using the BACKUP command before you continue.

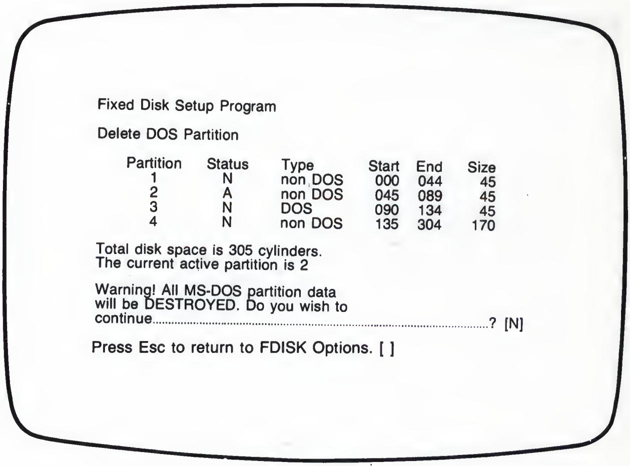
1. If you wish to continue working with MS-DOS after deleting the MS-DOS partition, you must insert the MS-DOS system diskette in drive A before proceeding. If you wish to continue working with some other operating system then you should either insert the system diskette of the appropriate operating system in drive A, or make the hard disk partition of that operating system active before deleting the MS-DOS partition.

2. Choose selection 3 from the FDISK Options menu by typing:

   3

   then press **ENTER**

   A video display similar to the following will appear:



```
Fixed Disk Setup Program

Delete DOS Partition

     Partition    Status    Type       Start  End    Size
         1          N       non DOS    000    044     45
         2          A       non DOS    045    089     45
         3          N       DOS        090    134     45
         4          N       non DOS    135    304    170

Total disk space is 305 cylinders.
The current active partition is 2

Warning! All MS-DOS partition data
will be DESTROYED. Do you wish to
continue.......................................? [N]

Press Esc to return to FDISK Options. [ ]
```

Fig. 2-11 Delete Partition Prompt

3. If you decide to cancel the operation press **ESC** and the FDISK Options menu will reappear. Conversely, if you have backed-up all your files and you wish to proceed with the deletion, type:

   Y

   then press **ENTER**

   and the MS-DOS partition is deleted and the displayed partition map is updated.

4. Press **ESC** to return to FDISK options [].

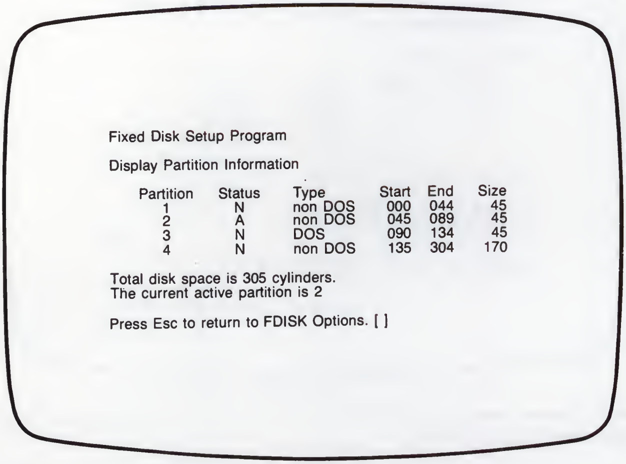
How to Display the Hard Disk Partition Map

If you type:

    4

then press **ENTER**

from the FDISK Options menu, a hard disk partition map similar to the following will be displayed:



```
Fixed Disk Setup Program

Display Partition Information

    Partition   Status    Type        Start   End    Size
        1         N       non DOS      000    044     45
        2         A       non DOS      045    089     45
        3         N       DOS          090    134     45
        4         N       non DOS      135    304    170

Total disk space is 305 cylinders.
The current active partition is 2

Press Esc to return to FDISK Options. [ ]
```

Fig. 2-12 Partition Map

The video display shows a typical partition map of a hard disk having four partitions.

The display shows for each set up partition

- The status of the partition; that is, whether or not that particular partition contains the operating system that will be booted when the system is rebooted from the hard disk. If so, it is designated the active partition (indicated by ''A''), while the other partitions are not active (indicated by ''N'').
- Whether or not the partition is type DOS or non-DOS.
- The beginning and end of the partition in terms of the first and last cylinder numbers.
- The size of the partition.

When you are ready to return to the FDISK Options menu, press **ESC**.

3. MS-DOS CONTROL KEYS AND EDITING FUNCTION KEYS

ABOUT THIS CHAPTER

This chapter describes the keystroke combinations that are required to execute control functions such as **ENTER** carriage return, line feed, break, etc. It also indicates the keystrokes that perform special editing functions. For further details of the latter refer to Chapter 8.

CONTENTS

INTRODUCTION 3-1

MS-DOS CONTROL KEYS 3-1

EDITING FUNCTION KEYS 3-4

INTRODUCTION

Other than enabling you to enter text in the manner of a typewriter, your keyboard is capable, under MS-DOS, of allowing you to enter other functions by using certain combinations of key-strokes. Such features fall into two groups: control keys and editing function keys.

MS-DOS CONTROL KEYS

MS-DOS control keys utilize a variety of key-stroke combinations. They are used to correct typing errors, abort command execution, etc. These functions are described in the following table. Note if you are using an Olivetti extended function key Keyboard the key combinations will have minor differences from those stated here (see your Installation and Operations Guide).

FUNCTION	KEY COMBINATION	MEANING
ENTER	↲ or **CTRL M**	The command line is terminated and command execution begins. Note that this key will be referred to as **ENTER** throughout this book.
Shift	↑ **(SHIFT)**	Activates the shifted value for a key (or gives a lower case value when **CAPS LOCK** is depressed).
Shift lock	**CAPS LOCK**	All alphabetic keys take upper case values. Press once and this key locks (toggles on). Press it again and it releases (toggles off).

▶

FUNCTION	KEY COMBINATION	MEANING
Keypad lock	**NUM LOCK**	Activates the numeric keys on the right hand keypad. Press once and this key locks (toggles on). Press it again and it releases (toggles off). The keys on the numeric key pad, as a default generate their cursor function.
Backspace	← or **CTRL H**	The last character entered is removed from the command line and the character erased from the screen.
TAB	**CTRL I** or ⇤ ⇥	The cursor is moved to the next eight-character TAB position.
Line Feed	**CTRL ENTER** or **CTRL J**	Moves the cursor to the start of the next screen line where you can continue to enter the line you are typing.
Escape	**ESC**	Cancels the current line and moves the cursor to the next line of the screen. A backslash (\) is displayed at the end of the cancelled line.

▶

FUNCTION	KEY COMBINATION	MEANING
Abort	**CTRL BREAK** or **CTRL C**	Aborts the current command. The command cannot be recommenced at the point of termination. If you wish to re-execute the command you must enter it again.
Suspend	**CTRL NUM LOCK** or **CTRL S**	Output to the screen is suspended. You can restart the display by pressing any other character key.
Echo Output	**CTRL PRT SC** or **CTRL P**	All output that is normally displayed on the screen is sent to the printer. Press the same keys again to terminate printer echo.
Print Screen	**SHIFT PRT SC**	Sends to the printer a copy of whatever is displayed on the screen. For graphic screens the MS-DOS command GRAPHICS has to be executed before pressing these keys (see Chapter 6).
System Reset	**CTRL ALT DEL**	Ends the current working session and reboots MS-DOS.

EDITING FUNCTION KEYS

These commands are entered by pressing a single key. Most of these commands use the function (F) keys.

When you press **ENTER** to execute a command, a copy of the command is kept in a special buffer called the source line. MS-DOS has a range of commands that you can use to enter a command line by copying some or part of the source line. For example if you wish to copy several files from drive B to drive A you might copy the first of those files by typing:

```
COPY B:MYFILE A:   ENTER
```

When you press **ENTER** this command would be written to the source line. If the next file you want to copy is called YOURFILE, then the character strings ''COPY B:'' and ''FILE A:'' can be copied from the source line using the function keys. The only part you need type in yourself is the string ''YOUR''.

The most common use of editing function keys, however, is in performing edit operations within a line of text when using the line editor EDLIN. A detailed description is therefore given in Chapter 8, but a brief description of each of the editing function keys is given in the following table:

COMMAND	KEY	FUNCTION
BACKUP	←	Erases one character from command line.
COPY1	**F1** or →	Copies one character from the source line to the command line.
COPYTO	**F2** then type a *character*	Copies all characters up to the first occurrence of *character* from the source line to the command line.

▶

COMMAND	KEY	FUNCTION
COPYLINE	**F3**	Copies all remaining characters in the source line to the command line.
SKIP1	**DEL**	Skip over one character in the source line.
SKIPTO	**F4** then type a *character*	Skip over characters in the source line up to *character*.
KILL	**ESC**	Terminates input and terminates the command line.
INSERT	**INS**	Enters/exits insert mode.
NEWTEMP	**F5**	Creates a new source line by copying the command line to the source line, but does not execute.

4. FILES AND DIRECTORIES

ABOUT THIS CHAPTER

This chapter describes how to manipulate files and directories, how you can access files in directories by means of paths and how to create and delete directories.

For further details of commands mentioned in this chapter refer to Chapter 6.

CONTENTS

HOW MS-DOS KEEPS TRACK OF YOUR FILES 4-1

HOW TO NAME YOUR FILES 4-1

WILD CARDS 4-2

THE ? WILD CARD 4-2

THE * WILD CARD 4-3

RESERVED DEVICE NAMES 4-4

HOW TO COPY YOUR FILES 4-5

HOW TO PROTECT YOUR FILES 4-6

HOW TO EXAMINE THE CONTENTS OF A FILE 4-6

DIRECTORIES 4-6

FILE NAMES AND PATHS 4-9

PATHNAMES 4-9

PATHS AND EXTERNAL COMMANDS 4-11

HOW TO DISPLAY YOUR CURRENT DIRECTORY 4-12

HOW TO CREATE A DIRECTORY 4-14

HOW TO CHANGE YOUR CURRENT DIRECTORY 4-14

HOW TO REMOVE A DIRECTORY 4-15

HOW TO CHECK THE VALIDITY OF YOUR FILES 4-15

HOW MS-DOS KEEPS TRACK OF YOUR FILES

The names of files are kept in directories on disk. These directories also contain information on the size of the files, their location on the disk, and the dates that they were created or updated. The directory you are working in is called your current directory.

An additional system area is called the File Allocation Table. It keeps track of the location of your files on the disk. It also allocates the free space on your disks so that you can create new files.

These two system areas, the directories and the File Allocation Table, enable MS-DOS to recognize and organize the files on your disks. The File Allocation Table is created on a disk when you format it with the MS-DOS FORMAT command. One empty directory is also created on it, known as the root directory.

HOW TO NAME YOUR FILES

A file name can comprise:

- One to eight characters (for legal characters see below). For example: NEWFILE.

- One to eight characters, followed by a period (.) and a one to three character file name extension. For example NEWFILE.EXE.

A file name may be made up of any of the following characters:

A-Z 0-9 $ & # ~

% ' ( ) - _

@ ^ { } !

Alphabetic characters within the file name can be entered in upper or lower case, but MS-DOS will translate lower case letters into upper case.

How you specify a file depends on which directory of which disk drive it resides:

- If the file is in the current directory on the default drive you need only specify the file name, for example:

```
MYFILE
```

- If you are using tree-structured directories (see the section entitled ''Directories'' later in this chapter) and the file is on the default drive but not in the current directory, then you need to specify a path; for example:

```
\MIDIR\MYFILE
```

Path names are described later in this chapter in the section entitled ''File Names and Paths''

- If the file is not on the default drive, you will need to specify the drive; for example:

```
B:MYFILE
```

WILD CARDS

Two special characters (called wild cards) can be used in file names and extensions: the asterisk (*) and the question mark (?). These special characters give you greater flexibility when using file names in MS-DOS commands.

THE ? WILD CARD

A question mark (?) in a file name or file name extension indicates that any character can occupy that position. For example, the MS-DOS command:

```
DIR TEST?RUN.EXE
```

will list all entries in the current directory on the default drive that begin with TEST, have any next character, end with the letters RUN, and have a file name extension of EXE. Here are some examples of files that might be listed by the above DIR command:

```
TEST1RUN.EXE
TEST2RUN.EXE
TEST6RUN.EXE
```

THE * WILD CARD

An asterisk (*) in a file name indicates that any valid character or sequence of valid characters can occupy that position in the file name. Any characters after the * are ignored by the system. For example:

```
DIR TEST*.EXE
```

will list all entries in the current directory on the default drive with file names that begin with the characters TEST and have an extension of EXE. Here are some examples of files that might be listed by the above DIR command:

```
TEST1RUN.EXE
TEST2RUN.EXE
TEST6RUN.EXE
TESTALL.EXE
```

The wild card designation *.* refers to all files in the current directory. Note that this can be very powerful when used in MS-DOS commands. For example, the command:

```
COPY A:*.* C:
```

copies all files on the current directory of the A: drive, regardless of file name, to the current directory of the C: drive.

Examples

To list all files named NEWFILE with any extension in the current directory on drive A, simply enter:

```
DIR A:NEWFILE.*
```

To list all file names that have less than or equal to six-characters plus an extension of .TXT in the current directory of the diskette in B, enter:

```
DIR B:??????.TXT
```

RESERVED DEVICE NAMES

MS-DOS treats device names specially, and certain words are reserved for the names of these devices. These names cannot be used as file names or extensions. Reserved names are as follows:

AUX: or COM1: Used when referring to input from or output to the built-in communications port.

COM2: Used when referring to input from or output to a second communications port.

CON: Used when referring to keyboard input or screen output.

PRN: or LPT1: Used when referring to the first parallel printer.

LPT2: or LPT3: Used when referring to the second and third parallel printers.

NUL: Used when you do not want to create a particular file, but the command requires an input or output file name.

The colon following the reserved device name is optional.

You can use a device name instead of a file name. Using "Redirection" (see Chapter 5) data can then be input from a device or output to a device, instead of a file.

HOW TO COPY YOUR FILES

You often need more than one copy of a file. The COPY command allows you to copy one or more files to another disk, to another file on the same disk or to another directory on the same disk (see the section entitled ''File names and Paths''). You can also give the copy a different name if you specify the new name in the COPY command.

You cannot make a copy of a file in the same directory unless you specify a different file name for the new copy.

For example,

```
COPY A:MYFILE.TXT B:MYFILE.TXT
```

will copy the file MYFILE.TXT on the diskette in drive A to a file named MYFILE.TXT on the diskette in drive B.

If you want to duplicate the file named MYFILE.TXT on the same disk, enter:

```
COPY A:MYFILE.TXT A:NEWNAME.TXT
```

You now have two copies of your file on drive A one named MYFILE.TXT and the other named NEWNAME.TXT.

You can also use the COPY command, along with the wild card feature, to copy groups of files from one disk to another, or to copy all files from one disk to another.

For example, the command

```
COPY A:*.EXE C:
```

copies all the .EXE from the current directory of the A: drive, to the current directory of the C: drive.

HOW TO PROTECT YOUR FILES

MS-DOS is a powerful and useful environment for processing your personal and business information. As with any information system, inadvertent errors may occur and information may be misused. If you are processing information that cannot be replaced or requires a high level of security, you should take steps to ensure that your data and programs are protected from accidental or unauthorized use, modification, or destruction. Simple measures you can take, such as removing your disks when they are not in use, keeping back-up copies of valuable information, write-protection, and installing your equipment in a safe place, can help you maintain the integrity of the information in your files.

HOW TO EXAMINE THE CONTENTS OF A FILE

There are two commands that enable you to examine a file. These are:

- The TYPE command that displays the contents of a text file.

- The HEXDUMP file that displays a file in hexadecimal.

If you want to print a file you can do this by means of the ">" I/O redirection symbol. For example enter:

```
TYPE myfile > PRN
```

will send the contents of myfile to the printer instead of the screen. For details about I/O redirection refer to Chapter 5.

DIRECTORIES

The names of your files are kept in directories on each disk. Each directory also contains information on the size of the files and the dates that they were created or last updated.

If your computer is used by several people, or if you are working on several different projects, the number of files in the directory can become large and unwieldy. Moreover, this will certainly be the case if you are using a hard disk as it is capable of storing a vast number of files. You may want your own files kept separate from those of a colleague: or, you may want to organize your programs into categories that are convenient to you.

MS-DOS allows you to organize the files on your disks into directories. Directories are a way of dividing your files into convenient groups of files. For example, you may want all your accounting programs in one directory and text files in another. Any directory can contain another directory, this is a sub-directory. This method of organizing your files is called a hierarchical directory structure.

A hierarchical directory structure can be thought of as a ''tree'' structure: directories are branches of the tree and files are the leaves, except that the ''tree'' grows downward; that is, the ''root'' is at the top. The root is the first level in the directory structure. It is the directory that is automatically created when you format a disk and start putting files in it. Any directory can contain files as well as sub-directories.

The tree or file structure grows as you create new directories for groups of files. Within each new directory, files can be added, or new sub-directories can be created.

It is possible for you to ''travel'' around this tree; for instance, you can find any file in the system by starting at the root and traveling down any of the branches to the desired file. Conversely, you can start where you are within the file system and travel towards the root.

The following figure illustrates a typical hierarchical directory structure:

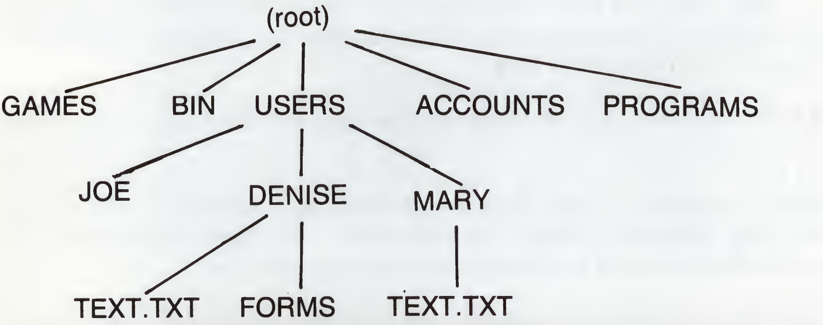


Fig. 4-1 A Sample Hierarchical Directory Structure

The root directory is the first level in the directory structure. You can create subdirectories from the root by using the MKDIR command. The root directory may also contain files.

In this example, five subdirectories of root have been created. These include:

- A directory of games, named GAMES.
- A directory of all external commands, named BIN.
- A USERS directory containing separate subdirectories for all users of the system.
- A directory containing accounting information, named ACCOUNTS.
- A directory of programs, named PROGRAMS.

Joe, Denise and Mary each have their own directories which are subdirectories of the USERS directory. Denise has a subdirectory under the \USERS\DENISE directory named FORMS. Denise and Mary have files in their directories, each named TEXT.TXT. Notice that Mary's text file is unrelated to Denise's.

This organization of files and directories is not important if you only work with files in your own directory; but if you work with someone else or on several projects at one time, the hierarchical directory structure becomes extremely useful. For example, you could get a list of the files in Denise's FORMS directory by entering:

```
DIR \USERS\DENISE\FORMS
```

Note that the back-slash (\) is used to separate directories from other directories and files. The first back-slash in a directory sequence represents the root directory.

To find out what files Mary has in her directory, you could enter:

```
DIR \USERS\MARY
```

FILE NAMES AND PATHS

When you use hierarchical directories, you must tell MS-DOS where the files are located in the directory structure. Both Mary and Denise, for example, have files named TEXT.TXT. Each will have to tell MS-DOS in which directory her file resides if she wants to access it. This is done by giving MS-DOS a pathname to the file.

PATHNAMES

A pathname is a sequence of one or more directory names followed, optionally, by a file name, each separated from the previous one by a back-slash (\).

The syntax of a pathname is:

[*drive*:][\][[*directory*][*directory*]...\]*filename*

or

[*drive*:][\][*directory*][*directory*]...

If a pathname begins with a backslash, MS-DOS searches for the file beginning at the root (or top) of the tree. Otherwise, MS-DOS begins at the users current directory and searches downward from there. For example, the path of Denise's TEXT.TXT file is \USERS\DENISE\TEXT.TXT. That is, if you are in directory JOE and you want to access the file named TEXT.TXT in directory DENISE you would use the pathname:

```
\USERS\DENISE\TEXT.TXT
```

If you were in directory USERS, however, you would access the same file using the pathname:

```
DENISE\TEXT.TXT
```

However, if you are already in directory DENISE you simply use the file name:

```
TEXT.TXT
```

If you are in directory MARY and you wish to make a copy of Mary's file TEXT.TXT in Denise's directory, but rename the copy COPY.TXT you would enter:

```
COPY TEXT.TXT \USERS\DENISE\COPY.TXT
```

MS-DOS provides special shorthand notations for the current directory and the parent directory (one level up) of the current directory:

.   MS-DOS uses this shorthand notation to indicate the name of the current directory in all hierarchical directory listings. MS-DOS automatically creates this entry when a directory is made

..   The shorthand name of the current directory's parent directory. If you enter

```
DIR ..
```

then MS-DOS will list the files in the parent directory of your current directory.

If you enter

```
DIR ..\..
```

then MS-DOS will list the files in the parent's parent directory.

(Note that the maximum length of a pathname is 63 characters).

PATHS AND EXTERNAL COMMANDS

External commands reside on disk as program files. They must be read from the disk before they execute.

When you are working with more than one directory, it is convenient to put all MS-DOS external commands into a separate directory so they do not clutter your other directories. When you issue an external command to MS-DOS, MS-DOS immediately checks your current directory to find that command. You must tell MS-DOS in which directory these external commands reside. This is done with the PATH command.

For example, if your current directory is named \BIN\PROG, and all MS-DOS external commands are in \BIN\COMMANDS, you must tell MS-DOS to choose the \BIN\COMMANDS path to find an external command. Enter the command:

```
PATH \BIN\COMMANDS
```

tells MS-DOS to search first your current directory and then the \BIN\COMMANDS directory for all commands. You only have to specify this path once to MS-DOS during your working session. The PATH command can be in AUTOEXEC.BAT If you want to know what the current path is, enter the PATH command without a parameter and the current value of PATH will be displayed.

You may also specify several paths. Refer to the PATH command in Chapter 6 for more information.

HOW TO DISPLAY YOUR CURRENT DIRECTORY

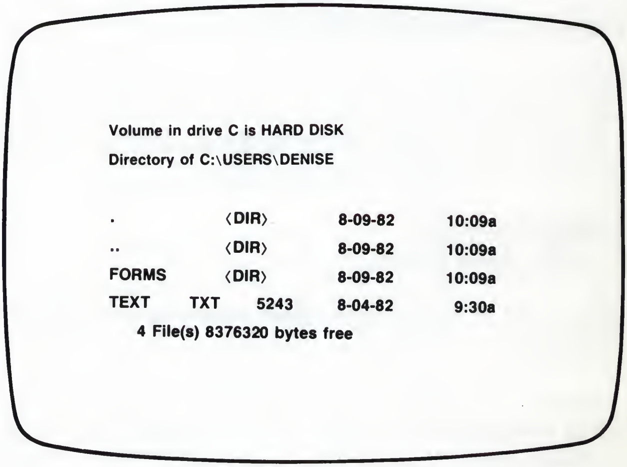
You can find out the name of the directory you are in by issuing the MS-DOS command CD or CHDIR (Change Directory) with no parameter. For example, if your current directory is \USERS\DENISE, when you enter:

```
CD
```

you will see:

```
C:\USERS\DENISE
```

This is the complete path of your current directory and comprises your current drive designation plus the current directory (\USERS\DENISE). If you now want to see what is in the \USERS\DENISE directory, you can issue the MS-DOS command DIR. The following is an example of the display you might receive for the DIR command for a subdirectory:



```
Volume in drive C is HARD DISK
Directory of C:\USERS\DENISE

.              <DIR>        8-09-82     10:09a
..             <DIR>        8-09-82     10:09a
FORMS          <DIR>        8-09-82     10:09a
TEXT      TXT       5243    8-04-82      9:30a
    4 File(s) 8376320 bytes free
```

Fig. 4-2 Subdirectory Example

Note that MS-DOS lists files and directories in this output. As you can see, DENISE has another directory in this tree structure named FORMS. The '.' indicates the current directory \USERS\DENISE, and the '..' is the shorthand notation for the parent directory USERS. TEXT.TXT is a file in the \USERS\DENISE directory. All of these directories and files reside on the disk in drive C.

Because files and directories are listed together (see previous display), MS-DOS does not allow you to give a subdirectory the same name as a file in that directory. For example, consider again the path USERS\DENISE\FORMS. FORMS is a subdirectory of DENISE, therefore you cannot create a file named FORMS in directory DENISE.

HOW TO CREATE A DIRECTORY

To create a subdirectory in your current directory, use the MD or MKDIR (Make Directory) command. For example, if you are in directory DENISE and you want to create a new directory named NEWDIR within your current directory, simply type:

```
MD NEWDIR
```

then press **ENTER**

and a new directory will exist in your tree structure under your current directory.

You can also create directories anywhere in the tree structure by specifying MD and then a path. For example, if your current directory is DENISE and you want to create a directory named SPECS in directory MARY you would type:

```
MD \USERS\MARY\SPECS
```

then press **ENTER**

MS-DOS will automatically create the . and .. entries in the new directory.

To create text files in the new directory, use either the Video File Editor (see Chapter 7) or the MS-DOS line editor EDLIN (see Chapter 8).

HOW TO CHANGE YOUR CURRENT DIRECTORY

To change your current directory to another directory simply issue the CD or CHDIR (Change Directory) command and supply a path. For example type:

```
CD \USERS
```

then press **ENTER**

to change the current directory to \USERS. You can specify any path after the command to "travel" to different branches and leaves of the directory tree. The command "CHDIR .." will always put you in the parent directory of your current directory (unless you are in root).

HOW TO REMOVE A DIRECTORY

To remove a directory from the tree structure, use the MS-DOS RD or RMDIR (Remove Directory) command. For example, to remove the directory NEWDIR from the current directory, type:

```
RD NEWDIR
```
then press **ENTER**

Note that the directory NEWDIR must be empty except for the . and .. entries before it can be removed; this will prevent you from accidentally deleting files and directories. You can remove any directory by specifying its path. To remove the \USERS\JOE directory, make sure that it has only the . and .. entries, then type:

```
RD \USERS\JOE
```
then press **ENTER**

If the directory is not empty an error is reported. To remove all the files in a directory (except for the . and .. entries), enter DEL and then the path of the directory. For example, to delete all files in the \USERS\DENISE directory, type:

```
DEL \USERS\DENISE
```
then press **ENTER**

and MS-DOS will display:

```
Are you sure? (Y/N)
```

Press **Y** and the files will be deleted. You cannot delete the . and .. entries (except by removing the directory).

HOW TO CHECK THE VALIDITY OF YOUR FILES

The CHKDSK command is used to check your disks for consistency and errors. The CHKDSK command analyzes the directories and the File Allocation Table on the disk that you specify. It then produces a status report of any inconsistencies, such as files which have a non-zero size in their directory but really have no data in them.

To check the disk in drive A type:

```
CHKDSK A:
```

then press **ENTER**

MS-DOS will display a status report and any errors that it has found. Moreover, if you include the flag switch **/V** in the command line:

```
CHKDSK A: /V
```

then the CHKDSK command displays messages as it runs and also shows the hidden files. You should run the CHKDSK command occasionally for each disk to ensure the integrity of your files.

5. ENTERING AND USING MS-DOS COMMANDS

ABOUT THIS CHAPTER

This chapter defines the syntax for a command, explains how commands can be grouped into batch files, and how the output from a command can be redirected to some device other than the VDU. It also describes the concept of ''piping'', whereby the output from one command becomes the input to another.

For further details of commands mentioned in this chapter refer to Chapter 6.

CONTENTS

COMMAND SYNTAX 5-1

PARAMETERS 5-2

INTERNAL AND EXTERNAL COMMANDS 5-6

ENTERING AND USING MS-DOS COMMANDS 5-6

EXTERNAL COMMAND SYNTAX 5-8

INFORMATION COMMON TO ALL MS-DOS COMMANDS 5-9

BATCH PROCESSING 5-10

HOW TO CREATE AND EXECUTE A BATCH FILE 5-11

THE AUTOEXEC.BAT FILE 5-14

BATCH FILE WITH REPLACEABLE PARAMETERS 5-15

INPUT AND OUTPUT 5-16

REDIRECTING YOUR OUTPUT 5-16

REDIRECTING YOUR INPUT 5-17

FILTERS 5-17

COMMAND PIPING 5-18

COMMAND SYNTAX

The general command format is defined as follows:

KEYWORD [*parameter*] ...

Where

SYNTAX ELEMENT	MEANING
KEYWORD	A one to eight character mnemonic that specifies the command to be executed. It must exclude any file name extension.
parameter	A parameter to the command defining the command action. The number of parameters depends on the command executed.

Remarks

If the KEYWORD is the name of an executable file, it may not reside on the default drive and directory. In this case the general command format can be extended (see the section on ''EXTERNAL COMMAND SYNTAX'').

PARAMETERS

Parameters are user-selected strings of alphabetic characters and of integers which can be optional. They are recognized by their position in the command line. The types of parameter are described in the following table:

PARAMETER TYPE	MEANING
filename or *directory*	Either a one to eight character string or a one to eight character string followed by a period (.) and a three character extension. A file name must be made up from the following characters: A-Z 0-9 $ & # ~ % ’ ( ) - — @` ^ { } ! Note: lower-case letters are transformed into upper case. For example: NEWFILE NEWFILE.TXT
filespec	[*drive*:]*filename* A file specifier (*filespec*) can be a file name with or without a drive specifier (*drive*). For example: NEWFILE.TXT B:NEWFILE.TXT

▶

PARAMETER TYPE	MEANING
pathname	[*drive*:][\][[*directory*][*directory*]...\]*filename* or [*drive*:][\][*directory*][*directory*]... a pathname may comprise: • a drive specifier followed by a colon, specifying the current directory of the specified drive; for example: C: • one or more directory names separated by backslashes specifying a directory path relative to the current directory of the default drive; for example: MARY\PROGRAMS\PASCAL • a drive specifier followed by one or more directory names separated from each other by backslashes specifying a directory path starting from the current directory of the specified drive; for example: C:MARY\PROGRAMS\PASCAL • a backslash specifying the root directory of the default drive: \ • a drive specifier followed by a colon and a backslash specifying the root directory of the specified drive; for example: C:\

▶

PARAMETER TYPE	MEANING
	• a backslash followed by one or more directory names separated from each other by backslashes specifying a directory path starting from the root directory of the default drive; for example: \USER\MARY\PROGRAMS\PASCAL • a drive specifier followed by a colon, a backslash and one or more directory names separated by backslashes specifying a directory path starting from the root directory of the specified drive; for example: C:\USER\MARY\PROGRAMS\PASCAL • any of the above definitions followed by a file name • a file name
drive	A single letter specifying the diskette drive or hard disk.
switch	An option which controls the way the command executes. It often takes the form of a single letter preceded by a slash. For example: **/P**
argument	Provides more information to an MS-DOS command. For example: ON or OFF

▶

PARAMETER TYPE	MEANING
volume label	One to eleven character string. Spaces may be included but not tabs. A volume label must be made up from the following characters: A-Z 0-9 $ & # ~ % ' ( ) - _ @ { } ! a-z Note: Lower case is converted to uppercase For example: Hard Disk -> HARD DISK

Nil Parameters

Some parameters are optional and take default values if they are not specified in the command line. For example, if you enter:

```
MODE COM1:11
```

then the last three parameters of the MODE COM command, which takes four parameters in all, take default values.

If default parameters are required in the middle of a command line a comma must be entered instead of each omitted parameter. For example, if you enter

```
MODE COM1:11,,,1
```

then the second and third parameters take default values. Note that the command line takes three commas. This is because the first one is required to separate the first parameter from the second, then the second and third commas replace the second and third parameters.

INTERNAL AND EXTERNAL COMMANDS

There are two types of MS-DOS command:

- Internal commands

- External commands

Internal commands are the simplest, most commonly used commands. You cannot see these commands when you do a directory listing on your MS-DOS disk; they are part of the command processor. Therefore they reside in memory whenever MS-DOS is loaded. When you enter these commands, they execute immediately. This class of command comprises:

BREAK
CHDIR (CD)
CLS
COPY
CTTY
DATE
DEL (ERASE)
DIR
ECHO
EXIT
FOR
GOTO
IF
MKDIR (MD)
PATH
PAUSE
PROMPT
REM
REN (RENAME)
RMDIR (RD)
SET
SHIFT
TIME
TYPE
VER
VERIFY
VOL

ENTERING AND USING MS-DOS COMMANDS

External commands reside on disk as program files. Any file name with a file extension of .COM, .EXE, or .BAT is considered to be an external command. They must be read from disk and loaded into memory before they can execute. With most of the external commands following execution they are removed from memory. However a few of these external commands (GRAFTABL, GRAPHICS, PRINT and SHARE) remain resident in memory after they have executed.

When you enter an external command, do not include its file name extension. External commands include:

ASSIGN
ATTRIB
BACKUP
CHKDSK
COMMAND
COMP
DEBUG
DISKCOMP
DISKCOPY
EDIT
EDLIN
EXE2BIN
FC
FDISK
FIND
FORMAT
GRAFTABL
GRAPHICS
GWBASIC
HEXDUMP
JOIN
LABEL
LINK
MODE
MORE
PRINT
RECOVER
RESTORE
SELECT
SHARE
SORT
SUBST
SYS
TREE

So for example FORMAT is the command file FORMAT.COM and ATTRIB is the executable file ATTRIB.EXE, but you call them as FORMAT and ATTRIB respectively. .EXE files have to be located in memory when they are loaded. Some .EXE files can be converted to .COM files using the MS-DOS utility EXE2BIN. .COM files are in memory image format and always load starting at location 100H in a memory segment, therefore .COM format is more compact and loads faster. Because all external commands reside on disk, you can create commands and add them to the system by writing programs in assembler or high level languages and compiling them. MS Compilers and the assembler MASM produce object code (.OBJ) files. These .OBJ files have to be linked, using the linker LINK (see Chapter 9). The LINK produces .EXE (executable) files. If the .EXE cannot be converted to .COM files the following error message appears:

File cannot be converted

Refer to Chapter 6 for more details on EXE2BIN.

When you specify an external command simply as KEYWORD then MS-DOS first looks in the default directory of the default drive, it then searches the paths set in the PATH variable of the environment, if the KEYWORD command file is not found, it cannot be executed and the following error message is output:

Bad command or filename

For external commands the general command format can be extended by preceding the KEYWORD with the Drive where the command file resides and/or the path leading to its directory.

EXTERNAL COMMAND SYNTAX

The general format of external commands is therefore defined as follows:

[*drive*:][*path*]**KEYWORD**[*parameter*]...

PARAMETER TYPE	MEANING
drive:	A one character drive specifier followed by a colon, specifying the drive where the KEYWORD is to be found.
path	{[\]directory[\directory]...\ | \} If the path consists of the root directory, only one backslash should be used, for example: C:\
filename or *directory*	Either a one to eight character string or a one to eight character string followed by a period (.) and a three character extension. A file name must be made up from the following characters: A-Z 0-9 $ & # ~ % ' ( ) - _ @ ^ { } !

PARAMETER TYPE	MEANING
filename or *directory* (cont.)	Note: lower-case letters are transformed into upper case. For example: NEWFILE NEWFILE.TXT
KEYWORD	A one to eight character mnemonic that specifies the command to be executed. It must exclude any file name extension.
parameter	A parameter to the command defining the command action. The number of parameters depends on the command executed. Refer to the preceding section ''PARAMETERS'' for more details of parameter types.

INFORMATION COMMON TO ALL MS-DOS COMMANDS

The following information applies to all MS-DOS commands:

- Commands are usually followed by one or more parameters.
- Commands and parameters may be entered in upper case or lower case, or a combination of both. MS-DOS will convert all lower case letters to upper case.
- Commands and parameters must be separated by delimiters. Space is usually used; for example:

  ```
  COPY A:MYFILE B:YOURFILE
  ```

  You can also use the comma (,), semicolon (;) or the equal sign (=) as delimiters in MS-DOS commands.

For brevity, this manual will use a space as the delimiter.

- When you are instructed to "Press any key", you can press any alphabetic (A-Z) or numeric (0-9) key.

- You must include the file name extension when referring to a file that already has a file name extension.

- You can abort commands that perform input/output by pressing **CTRL C** or **CTRL BREAK**.

- Commands take effect only after you have pressed **ENTER**.

- Wild cards (global file name characters) and device names (for example, PRN or CON) are not allowed in the names of any commands.

- When commands produce a large amount of output on the screen, the display will automatically scroll to the next screen. You can press **CTRL S** or **CTRL NUM LOCK** to suspend the display. Press any key to resume the display on the screen.

- MS-DOS control keys and editing function keys can be used when entering commands. Refer to Chapter 3 for a description of these keys.

- The default prompt from the command processor is the default drive designation plus a greater-than sign; for example, A>. You can change this prompt using the PROMPT command (see Chapter 6 for more details).

- Disk drives will be referred to as source drive and target drive. A source drive is the drive you will be transferring information from. A target drive is the drive you will be transferring information to.

BATCH PROCESSING

Often you may find yourself entering the same sequence of commands over and over to perform some commonly used task. With MS-DOS, you can put the command sequence into a special file called a batch file, and execute the entire sequence simply by entering the name of the batch file. "Batches" of your commands in such files are processed as if they were entered at the keyboard. Each batch file must be named with the .BAT extension, and is executed by entering the file name without its extension.

HOW TO CREATE AND EXECUTE A BATCH FILE

You can create a batch file by using the Video File Editor, the Line Editor (EDLIN) or the COPY command.

The MS-DOS command library contains a sub-set of batch processing commands. Among the more commonly used are REM and PAUSE. REM permits you to include remarks and comments in your batch files without these remarks being executed as commands. PAUSE prompts you with an optional message and permits you to either continue or abort the batch process at a given point.

Batch processing is useful if you want to execute several MS-DOS commands with one batch command, such as when you format and check a new diskette. For example, a batch file for this purpose might look like this:

```
REM This is a file to check new diskettes
REM It is named NEWDISK.BAT
PAUSE Insert new diskette in drive B
FORMAT B:
CHKDSK B:
```

Place an unprotected working copy of your system diskette in drive A.

Make sure that you have the prompt A> and proceed as follows to create the example batch file:

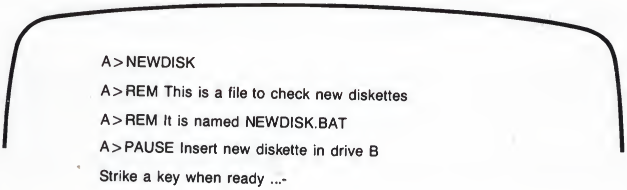
STEP	IF you enter...	THEN...
1	COPY CON: NEWDISK.BAT	the computer awaits input from the keyboard (CON:).
2	REM This is a file to check new diskettes	"REM This is a file to check new diskettes" is entered on the first line.

STEP	If you enter...	THEN...
3	REM It is named NEWDISK.BAT	"REM It is named NEWDISK .BAT" is entered on the second line.
4	PAUSE Insert new diskette in drive B	"PAUSE Insert new diskette in drive B" is entered on the third line.
5	FORMAT B:	"FORMAT B:" is entered on the fourth line.
6	CHKDSK B:	"CHKDSK B:" is entered on the fifth line.
7	**CTRL Z**	type **CTRL Z** the end-of-file character on the sixth line.
8	**ENTER**	Press **ENTER** the file creation is complete and the message: **1 File(s) copied** appears on the screen. The file NEWDISK.BAT is created on the system diskette.

To execute this batch file, simply enter the file name without the extension:

```
    NEWDISK
```

The result is the same as if each of the lines in the batch file were entered at the terminal as individual commands. That is, the first three commands are executed successively and the following messages are displayed on the screen:



```
A>NEWDISK
A>REM This is a file to check new diskettes
A>REM It is named NEWDISK.BAT
A>PAUSE Insert new diskette in drive B
Strike a key when ready ...-
```

Fig. 5-1 Sample Batch File Display

After striking a key, the diskette in drive B is formatted. You will then be asked if you wish to format another diskette. Following a negative reply, the diskette will be checked.

Remarks

1. Only the file name should be entered to execute the batch file. Do not enter the file name extension.

2. Do not name batch files with internal command names.

3. If you name batch files with external command names, the batch file will be executed in preference to the .EXE or .COM file.

4. If you press **CTRL C** while in batch mode, this prompt appears:

   ```
   terminate batch job (Y/N)?
   ```

   If you press **Y**, the remainder of the commands in the batch file are ignored and the system prompt appears.

   If you press **N**, only the current command is terminated and batch processing continues with the next command in the file.

5. If you remove the diskette containing a batch file being executed, MS-DOS prompts you to insert it again before the next command can be read.

6. The last command in a batch file may be the name of another batch file. This allows you to call one batch file from another, when the first is finished.

7. Input and Output can be redirected (the "<", ">") symbols. See later in this chapter for more information.

THE AUTOEXEC.BAT FILE

An AUTOEXEC.BAT file is a batch file that allows you to automatically execute programs when you start MS-DOS. Automatic Program Execution is useful when you want to run a specific package (for example, Multiplan) under MS-DOS, and when you want MS-DOS to execute a batch program automatically each time you start the system.

When you start MS-DOS, the command processor searches the MS-DOS system diskette for a file named AUTOEXEC.BAT. The AUTOEXEC.BAT file is a batch file that is automatically executed each time you start the system.

The AUTOEXEC.BAT file is created in exactly the same way as any batch file. It must, however, reside on the MS-DOS system diskette.

Example

If your AUTOEXEC.BAT file contains the following:

```
DATE
TIME
GWBASIC
```

then on initializing your system the date and time prompts will appear and the system will automatically enter GWBASIC.

BATCH FILES WITH REPLACEABLE PARAMETERS

You may want commands within a batch file to have replaceable parameters. For example, if your batch file contains a COPY command, you may wish to supply a different parameter to the COPY command each time you run the batch file. You can do this by specifying dummy (replaceable) parameters to the commands within the batch file. These parameters, named %0 to %9, can be replaced by values supplied when the batch file executes. For example, you may have created the following batch file named ''MYFILE.BAT'':

```
COPY %1.MAC %2.MAC
TYPE %2.TXT
TYPE %0.BAT
```

To execute this file you must enter the file name without extension, which is the value for parameter %0, followed by the replacement values for %1 and %2. For example, if you type:

```
MYFILE A:PROG1 B:PROG2
```
then press the **ENTER** key

then:

- %0 is replaced by ''MYFILE''
- %1 is replaced by ''A:PROG1''

- %2 is replaced by ''B:PROG2''

The effect is to execute the following sequence:

```
COPY A:PROG1.MAC B:PROG2.MAC

TYPE B:PROG2.TXT

TYPE MYFILE.BAT
```

Remarks

1. Up to 10 replaceable parameters (%0-%9) can be specified in this way. Refer to the SHIFT command if you wish to specify more than 10 parameters.
2. If you use the percent sign as part of a file name within a batch file, you must enter it twice. For example, to specify the file ABC%.EXE, you must enter it as ABC%%.EXE in the batch file.

INPUT AND OUTPUT

MS-DOS normally assumes that input comes from the keyboard and that output goes to the screen. However, the flow of command input and output can be redirected. Input can come from a file rather than the keyboard, and output can go to a file or to a printer instead of to the screen. In addition, ''pipes'' can be created that allow output from one command to become the input to another. Redirection and pipes are discussed in the next sections.

REDIRECTING YOUR OUTPUT

Most commands produce output that is sent to the screen. You can send this information to a file by using a greater-than sign (>) in your command. For example, the command:

```
DIR
```

displays a directory listing of the current directory on the screen. The same command can send this output to a file named MYFILES instead of the screen by designating the output file in the command line:

```
DIR >MYFILES
```

If the file MYFILES does not already exist, MS-DOS creates it and stores your directory listing in it. If MYFILES already exists, MS-DOS overwrites what is in the file with the new data.

Two greater-than signs (>>) can be used to tell MS-DOS to append the output of the command (such as a directory listing) to the end of a specified file. For example, the command:

```
DIR >>MYFILES
```

appends your directory listing to a currently existing file named MYFILES. If MYFILES does not exist, it is created.

REDIRECTING YOUR INPUT

It is often useful to have input for a command come from a file rather than from the keyboard. This is possible in MS-DOS by using a less-than sign (<) in your command. For example, the command:

```
SORT <NAMES >LIST1
```

sorts the file NAMES and sends the sorted output to a file named LIST1.

FILTERS

A filter is a command that reads your input, transforms it in some way, and then sends the output, usually, to the screen or to a file. In this way, the data is said to have been "filtered" by the program. Since filters can be put together in many different ways, a few filters can take the place of a large number of specific commands.

MS-DOS filters include FIND, MORE, and SORT. Their functions are described below:

FIND Searches for a constant string of text in a file.

MORE Takes standard output and displays it, one screen at a time.

SORT Sorts text.

Refer to Chapter 6 for more details of these commands.

You can see how these filters are used in the next section.

COMMAND PIPING

If you want to give more than one command to the system at a time, you can ''pipe'' commands to MS-DOS. For example, you may occasionally need to have the output of one program sent as the input to another program. A typical case would be a program that produces output in columns. It could be desirable to have this columnar output sorted.

Piping is done by separating commands with the pipe separator, which is the vertical bar symbol (|). For example, the command:

```
DIR | SORT
```

will sort your directory into alphabetical order. The vertical bar causes all output generated by the left side of the bar to be sent to the right side of the bar for processing.

Piping can also be used when you want to send output to a file. If you want your directory sorted and sent to a new file (for example, DIREC.FIL), you could enter:

```
DIR | SORT >DIREC.FIL
```

MS-DOS will create a file named DIREC.FIL on your default drive. DIREC.FIL contains a sorted listing of the directory on the default drive, since no other drive was specified in the command. To specify a drive other than the default drive, enter:

```
DIR | SORT >B:DIREC.FIL
```

This sends the sorted data to a file named DIREC.FIL on drive B.

A pipeline may consist of more than two commands. For example:

```
DIR | SORT | MORE
```

will sort your directory, show it to you one screen at a time, and put **-MORE-** at the bottom of your screen when there is more output to be seen. Press any typing key to see the next screen.

ABOUT THIS CHAPTER

This chapter gives the syntax and use of all the MS-DOS commands. The commands are presented in alphabetical order.

CONTENTS

INTRODUCTION 6-1

MS-DOS 3.1 AND NETWORKING 6-5

ASSIGN 6-8

ATTRIB 6-10

BACKUP 6-11

BREAK 6-14

CHDIR 6-15

CHKDSK 6-17

CLS 6-20

COMMAND 6-21

COMP 6-24

COPY 6-27

CTTY 6-34

DATE 6-35

DEL 6-38

DIR 6-40

DISKCOMP 6-41

DISKCOPY 6-45

ECHO 6-47

ERASE 6-47

EXE2BIN 6-48

EXIT 6-50

FC 6-51

FDISK 6-60

FIND 6-61

FOR 6-64

FORMAT 6-67

GOTO 6-70

GRAPHICS 6-72

GRAFTABL 6-75

GWBASIC 6-76

HEXDUMP 6-77

IF 6-80

JOIN 6-82

LABEL 6-87

MKDIR 6-88

MODE 6-89

MODE COM: 6-90

MODE n 6-92

MODE LPT 6-94

MORE 6-96

PATH 6-97

PAUSE 6-99

PRINT 6-100

PROMPT 6-104

RECOVER 6-107

REM 6-109

REN 6-109

RESTORE 6-111

RMDIR 6-115

SELECT 6-116

SET 6-118

SHARE 120

SHIFT 6-122

SORT 6-123

SUBST 6-125

SYS 6-129

TIME 6-130

TREE 6-132

TYPE 6-133

VER 6-134

VERIFY 6-135

VOL 6-136

WRITECHK 6-137

INTRODUCTION

The table below lists the commands included in this chapter and gives a brief description of each. It also tells you which commands are internal (I), which are external (E), and which it is not possible to use over a network (U).

COMMAND	CLASS	FUNCTION
ASSIGN	E	Instructs MS-DOS to route all requests for one drive to another drive.
ATTRIB	E	Sets or resets the read-only attribute of a file.
BACKUP	E	Creates a back-up of one or more disk files on a series of disks.
BREAK	I	Turns off and on the abort feature provided by **CTRL C** and **CTRL BREAK**
CHDIR	I	Changes the current directory.
CHKDSK	EU	Analyzes the contents of the disk in the specified or default drive.
CLS	I	Clears the screen.
COMMAND	E	Starts a new command processor
COMP	E	Compares the contents of a file or group of files with the contents of another file or group of files.
COPY	I	Copies one or more files to another file or to a device. Alternatively several files can be concatenated and copied to a destination file.
CTTY	I	Changes the input/output console from which you issue commands.

COMMAND	CLASS	FUNCTION
DATE	I	Displays and sets the date known to the system.
DEL	I	Deletes the specified file(s).
DIR	I	Lists the requested directory entries.
DISKCOMP	E U	Compares the contents of two diskettes of the same type.
DISKCOPY	E U	Copies the contents of a diskette in one drive onto another diskette.
ECHO	I	Turns the batch file echo feature off and on. Text input as a parameter will be output to the standard output device.
ERASE	I	Is the same as DEL.
EXE2BIN	E	Converts executable files to binary format.
EXIT	I	Exits from a secondary command processor and returns to a parent program or command processor.
FC	E	Compares the Contents of two files.
FDISK	E	Sets up the MS-DOS partition for the hard disk.
FIND	E	Searches for a specific string of text in a file or files.
FOR	I	Allows iterative execution of MS-DOS commands.
FORMAT	I	Formats a disk to receive MS-DOS files.

▶

COMMAND	CLASS	FUNCTION
GOTO	I	Jumps to a specified position in a batch file.
GRAPHICS	E	Enables graphics currently displayed on the screen to be printed on a compatible printer, along with any text when the **SHIFT PRT SCR** keys are pressed.
GWBASIC	E	Enters the MS GW-BASIC interpreter.
HEXDUMP	E	Displays the contents of a file, byte by byte, in hexadecimal.
IF	I	Causes conditional execution of a command in a batch file.
JOIN	E U	Joins a disk drive to an empty directory on another drive to produce a single directory structure.
LABEL	E U	Creates, changes or deletes a disk volume label.
MKDIR	I	Creates a directory.
MODE	E	Sets the monitor mode, serial transmission and printer environments.
MORE	E	Sends output to the terminal one screen at a time.
PATH	I	Sets a command search path.
PAUSES	I	Pauses for input in a batch file.
PRINT	E U	Queues text files for background printing.
PROMPT	I	Sets the MS-DOS command prompt.

COMMAND	CLASS	FUNCTION
RECOVER	E U	Recovers a file or an entire disk containing faulty blocks.
REM	I	Displays a remark during the execution of a batch file.
REN[AME]	I	Renames files.
RESTORE	E	Restores a number of files from backup disks. The backup disks must have been created using the BACKUP command.
RMDIR	I	Removes an empty sub-directory.
SELECT	E	Copies your MS-DOS diskette to create a working copy for your selected country and keyboard.
SET	I	Assigns one string value in the environment to another key string; for use in programs or batch files.
SHARE	E	Installs MS-NET file sharing and lock—ing.
SHIFT	I	Increases the number of replaceable parameters in batch processing.
SORT	E	Sorts data alphabetically, in forward or reverse order.
SUBST	E	Substitute a dummy drive specifier for a pathname.
SYS	E U	Updates the specified disk with the system files IO.SYS and MSDOS.SYS, which come from the default drive.

COMMAND	CLASS	FUNCTION
TIME	I	Displays and sets the system time.
TREE	E	Displays all the directories and paths on the specified drive. It also has an option to list the files in each directory.
TYPE	I	Displays the contents of the specified file on the video screen.
VER	I	This command displays on your screen the version number of the MS-DOS system you are using.
VERIFY	I	Verifies writes to disk.
VOL	I	Displays the volume label of the disk in the specified or default drive.
WRITECHK	E	Set write protection for 48 tpi diskettes in a High Capacity drive.

MS-DOS 3.1 AND NETWORKING

MS-DOS 3.1 supports networking using the MS-Network extension software.

The following command is specific to networking and only useful only when networking is active:

COMMAND	DESCRIPTION
SHARE	This program loads, then terminates, but stays resident in the Random Access Memory. It installs the file-sharing/record-locking mechanism.

The following commands are useful for networking:

COMMAND	**DESCRIPTION**
ATTRIB	This command sets or resets the read-only attribute of a file or displays the attributes of that file. If any application opens a file with read/write permission, ATTRIB can set the file to read-only, allowing certain application programs to be run and shared over the network in compatibility mode.
COPY	This command can be used to copy files from a network disk to your own or to another network disk.
DIR	This command can be used to display information about files on network disks.

Most MS-DOS commands can be used over the network. But do not use:

COMMAND	**Result...**
CHKDSK	the error message: **Cannot CHKDSK a Network drive.** If you suspect a problem contact the Network Manager.
DISKCOMP	the error message: **Cannot DISKCOMP to or from a Network drive.** Use COMP *.* for each relevant directory instead.

COMMAND	Result...
DISKCOPY	the error message: **Cannot DISKCOPY to or from a Network drive.** Use COPY *.* for each directory instead.
FDISK	the error message: **Cannot FDISK a Network Drive.**
FORMAT	the error message: **Cannot FORMAT a Network Drive.** The Network Manager can STOP the server, do FORMAT then RESTART the server.
JOIN	the error message: **Cannot JOIN a Network Drive.**
LABEL	the error message: **Cannot LABEL a Network Drive.**
PRINT	the error message: **Cannot use PRINT use NETPRINT.** Do as it suggests, use NETPRINT to print files over the network.
RECOVER	the error message: **Cannot RECOVER to a network drive.** The Network Manager can STOP the server, do RECOVER, then RESTART the server.
SUBST	the error message: **Cannot SUBST to a network drive.**
SYS	The error message: **Cannot SYS to a network drive.** The Network Manager can STOP the server, do SYS then RESTART the server.

ASSIGN

Instructs MS-DOS to route all requests for one drive to another drive.

Classification

External

```
[ d:][path] ASSIGN [drive1=drive2]...
```

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where ASSIGN is to be found.
path	Specifies the directory where ASSIGN is to be found.
drive1	The letter of the drive whose requests are to be rerouted.
drive2	The letter of the drive to which requests for drive1 are to be rerouted.

Characteristics

Following execution of the ASSIGN command MS-DOS converts all references for *drive1* to *drive2*.

Note that only the drive letter should be entered in the command line. Do not enter the colon.

If you enter the ASSIGN command without parameters all current assignments will be reset.

Examples

IF you enter...	THEN...
ASSIGN A=C B=C	all requests to drives A or B will be rerouted to drive C.
ASSIGN	all assignments are reset.

Remarks

Never ASSIGN drives and then use the following commands or unpredictable results will occur and/or error messages will be displayed:

BACKUP DISKCOMP DISKCOPY FORMAT JOIN LABEL PRINT RESTORE SUBST

ATTRIB

Sets or resets the read-only attribute of a file.

Classification

External

[*d*:][*path*] **ATTRIB [+R|-R]** *pathname*

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where ATTRIB is to be found.
path	Specifies the directory where ATTRIB is to be found.
pathname	The pathname of the files you want to reference. Wildcard characters (* and ?) can be used in the filename.

Characteristics

+R sets the read-only attribute of a file. -R disables read-only mode

To display the attribute of files enter ATTRIB pathname.

BACKUP

Creates a back-up of one or more disk files on a series of disks. The source is usually a hard disk, but can be a floppy disk. The target is usually a floppy disk, but can be a hard disk. However the source and the target must be different drives.

Classification

External

[*d*:][*path*] **BACKUP** *source-drive*:[*pathname*] *target-drive*: **[/S] [/M] [/A] [/D:** *date*]

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where BACKUP is to be found.
path	Specifies the directory where BACKUP is to be found.
source-drive	The disk drive to be backed up.

▶

SYNTAX ELEMENT	MEANING
pathname	The files you wish to backup. If you enter only the source-drive specifier then only those files in the current directory are backed up. If you specify a path terminating in a directory name then all files in that directory will be backed up. If the path terminates in a file name (or a group of file names specified using wild card characters) only the specified file(s) will be backed up.
target-drive	The disk drive in which the backup copy is to be made.
/S	The files contained in the subdirectories are to be backed up as well as those in the specified directory (or current directory if no directory is specified). This includes files at all directory levels below the specified directory.
/M	Only those files in the specified directory that have been modified or created since the last backup will be backed up.
/A	The specified files will be added to the disk already inserted in the diskette drive. If /A is not specified you will be prompted to insert a diskette once the BACKUP program is in memory.
/D:*date*	Only those files in the specified (or current) directory that have been created or modified since the specified date are to be backed up. Refer to the DATE command for valid date formats.

Characteristics

Once you have entered the BACKUP command a prompt will be issued asking you to insert a target diskette (unless you included /A in the command line). You must use MS-DOS formatted diskettes. Any files that already existed on the target diskette will be deleted unless you used the /A option. Once the target diskette is full you will be prompted to insert another target diskette. Be sure to label each diskette as the order will be important when you restore your backup to hard disk.

As each file is backed up its name is displayed on the screen.

The exit code is set by the BACKUP command as follows:

- 0 Normal completion.
- 1 No files found.
- 2 Some files not backed up due to file sharing conflicts.
- 3 Command execution terminated by the user.
- 4 Command execution terminated due to an error.

The exit code can be used by the batch processing IF command.

Note

The files on the backup diskettes cannot be used except for restoring to the source-drive using the RESTORE command.

Examples

IF you enter...	THEN...
BACKUP C:*.COM A:	Each file with the .COM extension in the current directory of the hard disk drive C is backed up onto a series of diskettes in drive A.

▶

IF you enter...	THEN...
BACKUP C:*.* A:/S	All files on the hard disk drive C will be backed up onto a series of diskettes in drive A.
BACKUP C:MYDIR\MYFILE A: /A	The file named MYFILE in the directory MYDIR is added to the backup diskette in drive A.
BACKUP *.* A: /M	All files in the current directory on the hard disk that have been created or modified since the last backup was made are backed up onto a series of diskettes in drive A.
BACKUP C:*.* A: /D:01-01-84	All files in the current directory on drive C that have been created or modified since 1 January 1984 are backed up onto a series of diskettes in drive A.

BREAK

Turns off and on the abort feature provided by **CTRL C** and **CTRL BREAK**.

Classification

Internal

BREAK [ON | OFF]

Characteristics

MS-DOS normally checks **CTRL C** only for input/output operations involving screen, keyboard and printer. **CTRL C** will not normally abort other functions, such as the assemble or compile operations. Specifying **BREAK ON** makes **CTRL C** effective for other functions.

If you specify BREAK without a parameter the current BREAK setting is displayed.

CHDIR

Changes the current directory; displays the name of the current directory.

Classification

Internal

Syntax 1

CHDIR [*drive*:][*path*]

Syntax 2

CD [*drive*:][*path*]

Where

SYNTAX ELEMENT	MEANING
drive	The letter of the drive where the new directory is to be found.
path	A path that terminates with the name of the directory you wish to enter.

Characteristics

Use CHDIR (or CD) with a path to change to any valid directory.

To change to the parent directory of your current directory enter:

```
CHDIR ..
```

Used without a parameter CHDIR displays the full path and name of your current directory. This feature is useful if you have forgotten the name of the directory in which you are working.

To change to the root directory enter:

```
CHDIR \
```

Examples

IF you enter...	THEN...
CHDIR \BIN\USER\JOE\FORMS	MS-DOS puts you in the directory \BIN\USER\JOE\FORMS
CHDIR ..	MS-DOS puts you in the parent of the current working directory (in the above example, \BIN\USER\JOE)

CHKDSK

Analyzes the contents of the disk in the specified or default drive.

Classification

External, Non-network

[*d*:][*path*] **CHKDSK** [*drive*:][*dir-path*][*filename*] [/**F**] [/**V**]

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where CHKDSK is to be found.
path	Specifies the directory where CHKDSK is to be found.
drive	The drive containing the disk to be checked.
dir-path	The path to the directory to be checked.
filename	A file or group of files specified using wild card characters. A status report for the individual files will be displayed, if they are non-contiguous.
/F	CHKDSK tries to correct any errors it finds.
/V	CHKDSK displays status messages for each directory, subdirectory and each file specified.

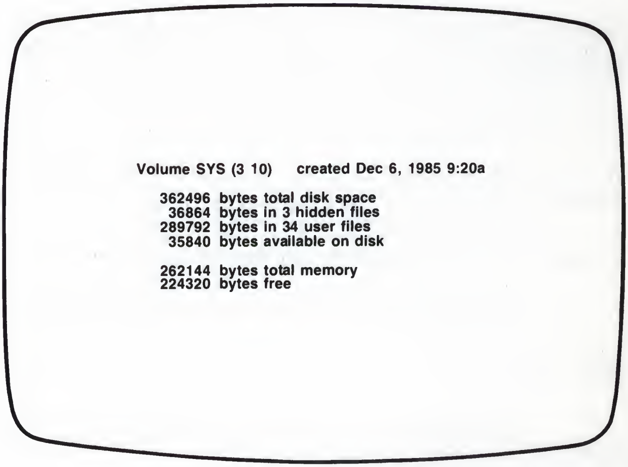
Characteristics

CHKDSK produces a status report on the File Allocation Tables, Directories and Files of the disk. If you specify the **/F** switch, CHKDSK attempts to correct any errors it finds. If you specify the **/V** switch, CHKDSK displays messages while it is running and will also list the hidden files.

If no parameter is specified then the disk in the default drive is checked.

Example

The following is an example of a CHKDSK status report:



```
Volume SYS (3 10)    created Dec 6, 1985 9:20a

   362496 bytes total disk space
    36864 bytes in 3 hidden files
   289792 bytes in 34 user files
    35840 bytes available on disk

   262144 bytes total memory
   224320 bytes free
```

Fig. 6-1 CHKDSK Example

Remarks

To redirect CHKDSK output, use the > redirection symbol followed by the name of a file. Error messages will be sent to the file specified. Do not use the **/F** switch if you redirect CHKDSK output.

If you use the **/F** switch CHKDSK tries to fix any errors found in the directory, file or file allocation table (FAT). When errors are found, due to lost clusters CHKDSK asks you ''Convert lost chains to files (Y/N)?''. If you reply **Y** then press **ENTER** CHKDSK recovers each cluster in the disconnected chain to a file called FILEnnnn, in the root directory of the specified drive (where nnn starts at 0000 and increases by 1 for each lost chain). See Appendix D ''ERROR MESSAGES'' for a list of all the messages CHKDSK outputs.

Space on diskettes is allocated in clusters. Diskettes that have had a lot of file creation and deletion activity become fragmented, because clusters are not allocated sequentially. The first free cluster found is the next cluster allocated regardless of its location on the diskette.

A fragmented diskette can cause poor performance due to delays involved in reading or writing a file.

CHKDSK will display one of the following messages:

filename contains non-contiguous blocks

or

All specified file(s) are contiguous

If the first message is output and you are experiencing poor disk performance, use the COPY command to copy all the files in the directory to a newly formatted diskette. Then use the copy rather than the original.

Use the command CHKDSK *.* to find out whether the files in the current directory are contiguous or not.

CLS

Clears the screen.

Classification

Internal

```
CLS
```

Characteristics

All data on the display screen is cleared. The cursor is moved to the upper left hand corner (the home position). This command has no effect on memory or files.

COMMAND

Starts a new command processor

Classification

External

[*d*:][*path*] **COMMAND** [*drive*:*shellpath*] [*cttydev*] [**/P**]
[**/C** *command-string*]

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where COMMAND is to be found.
path	Specifies the directory where COMMAND is to be found.
drive	A single letter drive-name of the drive containing COMMAND.COM, which is assigned to COMSPEC in the new environment.
shellpath	The pathname of the directory containing COMMAND.COM, which is assigned to COMSPEC in the new environment.
cttydev	An alternative device for standard input and output (see CTTY command for more details).
/P	This switch makes this copy of COMMAND.COM permanent. It is not possible to exit to the primary command processor without re-booting the system.
/C *command-string*	This switch causes the *command-string* to be passed to the new command processor for execution and then exit to the original command processor. This must be the last switch if used; everything following is part of the *command-string*.

Characteristics

COMMAND.COM is loaded into memory in two parts: the transient part and the resident part. Some application programs write over the transient part of the command processor when they run. When this happens the resident part looks into the COMSPEC variable in the environment, to find the command processor file, which is defined as:

```
COMSPEC = drive:shellpath
```

typically

```
COMSPEC = A:\COMMAND.COM
```

You can use SET without any parameters to examine the environment. (See SHELL in Appendix C for details on loading the top level command processor). One application of this is to install a copy of COMMAND.COM on Virtual Disk using (VDISK); calling it D:, for example. Then to issue the following call to invoke the second command processor.

```
D:\COMMAND D:\ /P
```

Another application is to call COMMAND without any parameters or only with *drive:shellpath*. This invokes a secondary command processor (a child), which inherits the parent command processor's environment and prompt. If this environment and/or prompt is modified, these changes are only known to the child processor and its applications. Using EXIT (see EXIT command in this chapter), reinstates the parents environment. For example at the **A>**prompt enter:

```
COMMAND
```

The computer will respond:

```
Microsoft MS-DOS

Version 3.10, etc

A>
```

enter:

```
PROMPT time=$T$G
```

the computer will respond with the prompt incorporating the time known to the computer:

time=17:10:11.80>

If you now enter:

```
EXIT
```

the computer exits to the parent processor which has the prompt:

A>

You call batch files in the following way:

COMMAND /C *batchfilename* [*parameter*...]

This syntax enables the original parent batch file to call a child batchfile with *batchfilename* and the command processor as a subroutine. Upon completion of the child batchfile, there is an automatic exit to the parent batchfile and command processor.

COMP

Compares the contents of a file or group of files with the contents of another file or group of files. This is useful for checking the results of a COPY operation.

Classification

External

[*d*:][*path*] **COMP** [*pathname1* [*pathname2*]]

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where COMP is to be found.
path	Specifies the directory where COMP is to be found.
pathname1	The file or group of files (specified using wild card characters) that the file or files specified by *pathname2* are to be compared with. If the path terminates in a directory name, all files in the specified directory are compared. If the path terminates in a drive specifier the files in the current directory of that drive are compared.
pathname2	The file or group of files (specified using wild card characters) to be compared with the file or group of files specified by *pathname1*. If the path terminates in a directory name or drive specifier, only the files with the same file name as those specified by *pathname1* will be compared.

Characteristics

The files you wish to compare may be on the same or different drives, or in the same or different directories.

If you enter the command without parameters, or if you omit the second parameter, you will be prompted for the missing parameters.

As the COMP command proceeds it displays the files and paths of the files currently being compared. An error message is issued if a specified directory path is invalid, or the two files to be compared are different in size, or a file specified by *pathname2* cannot be found.

If the same location in two files contains information which does not match, a message is issued indicating the offset (in bytes) within the files and the contents of the bytes for each file. If ten such mismatches are found the comparison terminates and issues an appropriate message.

If, at the end of a comparison, the *end-of-file* marker cannot be found, the following message will be displayed:

EOF mark not found

This is necessary as some applications create files that are always recorded in multiples of 128 bytes, although the data actually occupies a few bytes less than stated in the directory. The COMP command may therefore find compare errors in the last 128 bytes beyond the last actual data byte. The above message therefore indicates that the actual data in the two files matches.

Following a successful comparison COMP displays a message indicating that the files compare and continues comparing the next pair of files. After all the specified files have been compared the following prompt appears:

Compare more files (Y/N)?

Press **N** to terminate the command, or **Y** if you wish to compare more files. If you press **Y** you will be prompted for the files you wish to compare.

Examples

IF you enter...	THEN...
COMP A:*.LST B:*.CPY	all the files on drive A with the extension LST are compared with the files of the same name but with extension CPY on drive B.
COMP A:*.LST C:	all the files on drive A with the extension LST are compared with the files of the same name on the current directory of drive C.

COPY

Copies one or more files. Alternatively several files can be concatenated and copied to a destination file.

Classification

Internal

COPY [**/A**|**/B**]*pathname1* [**/A**|**/B**][+*pathname2*[**/A**|**/B**]...] [*pathname* [**/A**|**/B**]][**/V**]

Where

SYNTAX ELEMENT	MEANING
pathname1	The path of the file to be copied (excluding the drive only if the file is on the default drive, excluding the directory path only if the file is in the current directory).
pathname2	The path of any file to be concatenated with the file in pathname1 (excluding the drive specifier only if the file is in the default drive, excluding the directory path only if the file is in the current directory).
pathname	The path of the target directory or file (excluding the drive to place the file in the default drive, or the file name to retain that given in pathname1).
/V	Verify the target file by a read after a write and COPY then compares this data. An error message is output if this comparison fails

Characteristics

COPY either makes a copy of a file or concatenates two or more files. The resulting target file may have the same name as the first source file (so long as it is in another directory), or a different name.

To copy a file and retain its name, use the following syntax:

COPY *pathname1* [*pathname*]

where *pathname* gives the drive and/or directory in which the file is to be placed.

Omit the *pathname* parameter to place the copy in the current directory of the default drive. Specify simply a drive to place the copy in the current directory of that drive. If you specify a directory path, make sure that you end the path with the name of an existing directory.

Two files cannot have the same name in the same directory: if you try to copy a file onto itself you will get an error message.

To copy a file and give the copy a different name, use the following syntax:

COPY *pathname1 pathname*

where *pathname* gives the name of the file in which the copy is to be placed.

Omit drive and directory names from *pathname* to place the file in the current directory of the default drive. Omit just directory names from *pathname* to place the file in the current directory of the specified drive. Because you are renaming the file, the source and target directories may be the same.

If you identify in *pathname* a file that does not exist, MS-DOS creates it for you. If the file already exists, its previous contents are destroyed in the copy operation.

To concatenate two or more files, use the following syntax:

COPY *pathname1* + *pathname2* [+ *pathname3*]... [*pathname*]

Combine the names of each file to be concatenated with the plus sign (+). COPY appends each file in turn to the previous one.

The result of the concatenation is a single file. This file is given the drive, directory, name and extension specified in pathname, if this is absent, the resulting file replaces the file specified in *pathname1* (that is, the first file to be concatenated). If only the drive is supplied, a new file with the name of the file specified in *pathname1* is copied into the drive's current directory. If drive and directory names only are supplied, a new file with the name given in *pathname1* is copied into the specified subdirectory. If only the name and extension are supplied, the file is placed in the current directory of the default drive.

Examples

IF you enter...	**THEN...**
COPY B:SECRETS	the file SECRETS is copied from drive B to the default drive.
COPY *.* B:	all files in the current directory on the default drive are copied to drive B.
COPY \DOCS\SECRETS B:INFO	the file SECRETS is copied to the file INFO in the current directory of drive B (or the subdirectory INFO in the current directory if it exists).
COPY \DOCS\SECRETS \DOCS\NEWS	the file SECRETS is copied to the file NEWS (or the subdirectory NEWS if it exists).
COPY INFO+NEWS+VIEWS ALL.LST	the files NEWS and VIEWS are appended to the file INFO, and the resulting concatenation is copied to the file ALL.LST.

▶

IF you enter...	THEN...
COPY ALL.LST + NEWS	the file NEWS is appended to the file ALL.LST, resulting in an enlarged ALL.LST.
COPY *.LST COMBIN.PRN	all files with the extension .LST are concatenated, and the result is placed in COMBIN.PRN.
COPY *.LST + *.REF *.PRN	each file with the extension .REF is appended to the file with the same name but the extension .LST, and each resulting file is given the extension .PRN.
COPY *.LST + *.REF COMBIN.PRN	all files with the extension .LST then all files with the extension .REF are placed in COMBIN.PRN.
COPY ALL.LST + *.LST	all files with the extension .LST, with the exception of ALL.LST, are appended to ALL.LST.
COPY PROG.COM/B + ERRS.TXT/A	the text file ERRS.TXT is appended to the binary file PROG.COM, leaving the result in the binary file PROG.COM.

Remarks

As the examples show, you can use the wild cards * and ?, to do both simple copy operations (where no target file is specified) and file concatenation.

When wild cards are present in two or more source parameters combined with the concatenation symbol (+), the result is a single target file. However if the target parameter itself contains a wild card, a series of concatenated files are produced.

Note that the penultimate example shows the correct way of concatenating files where one of the source files is also the target file. Had the command COPY *.LST ALL.LST been entered, the previous contents of ALL.LST would have been destroyed and the following message would have appeared:

Content of destination lost before copy

You may use the following reserved device names in place of standard file name parameters (the colons are optional):

```
AUX:        LPT2:
COM1:       LPT3
COM2:       NUL:
CON:        PRN:
LPT1:
```

For example, to copy text that you are about to enter from the keyboard into a file, use the following format:

```
COPY CON: filename
```

You can then enter text directly into the file you have named, terminating your input by pressing **CTRL Z** followed by **ENTER**.

The parameters **/A** and **/B** shown in the syntax of the COPY command apply when you wish to regulate the amount of data to be copied. The following table shows the effect of **/A** and **/B** on files to which they are attached and to all remaining files in the command until another parameter is found.

IF you enter...	WITH...	THEN...
/A	a source file	the file is regarded as a text (ASCII) file, and its contents copied up to but excluding the first end-of-file character (**CTRL-Z**). This is the default.
/A	a target file	the file is regarded as a text (ASCII) file, and an end-of-file character (**CTRL-Z**) is added as its last character. This is the default.
/B	a source file	the file is regarded as a binary file, and the entire file including any number of end-of-file characters is copied.
/B	a target file	the file is regarded as binary, and no end-of-file character (**CTRL-Z**) is added.

The default value is **/A** when you are using COPY to concatenate files, **/B** when you are using COPY simply to copy files.

See the last example COPY PROG.COM/B+ERRS.TXT/A which shows the use of **/A** and **/B** to append a file of error messages to a program file. The default for concatenated files being **/A** the **/B** attached to the program file is obligatory. The **/A** must then be attached to the text file in order to cancel the previous **/B** parameter.

CTTY

Changes the input/output console from which you issue commands.

Classification

Internal

CTTY *device*

Where

SYNTAX ELEMENT	MEANING
device	The reserved name of the device you wish to use.

Characteristics

A suitable terminal must be connected to the device port. Command Input/ Output is passed to the alternate terminal. The CTTY CON command must be entered at the alternate terminal to restore input/output back to the normal console.

Examples

IF you enter...	THEN...
CTTY AUX:	command I/O is moved to the device attached to the RS-232-C.
CTTY CON:	command I/O is returned to the console.

Note

You must use MODE to initialize the device before use. There are many programs that do not use MS-DOS for input and/or output, but use the BIOS or hardware ports. The CTTY command will have no effect on these programs. CTTY will only affect programs that use MS-DOS.

DATE

Displays and sets the date known to the system.

Classification

Internal

Syntax 1 USA

DATE [*mm-dd-yy*]

or

Syntax 2 Europe

DATE [*dd-mm-yy*]

or

Syntax 3 Japan

DATE [*yy-mm-dd*]

Where

SYNTAX ELEMENT	MEANING
mm	The one- or two-digit identifier of the month (1-12).
dd	The one- or two-digit identifier of the day (1-31).
yy	The two- or four-digit identifier of the year (80-99 or 1980-2099).

Characteristics

The syntax depends on the COUNTRY setting in CONFIG.SYS.

Separate month, day and year entries by either hyphens (-) or slashes (/).

If you leave out the parameter, DATE prompts you as in the following example:

```
Current date is Tue 11-5-85
Enter new date:
```

Enter the date in the correct format, without entering a value for the day of the week. To accept the current date simply press **ENTER**.

Example

IF you enter...	THEN...
DATE 1-2-84	2nd January 1984 is established as the current date.
DATE	the DATE program prompts you to enter the date.

Remarks

If the values or separators you enter are not valid, DATE displays the message:

```
Invalid date
Enter new date:
```

DATE then waits for you to enter a valid date.

DEL

Deletes the specified file(s).

Classification

Internal

Syntax 1

DEL *pathname*

Syntax 2

ERASE *pathname*

Where

SYNTAX ELEMENT	MEANING
pathname	The specification of the file(s) to be deleted (excluding the drive to delete files in the default drive, excluding the directory path to delete file(s) in the current directory).

Characteristics

You may use the wild cards * and ? in the file name and extension.

To delete all the files in a directory enter the wild cards "*.*". Alternatively, enter a path ending in a directory. In these cases, MS-DOS prompts you to confirm your choice as follows:

Are you sure (Y/N)?

Press **Y** to carry out the deletion, or **N** to return to the MS-DOS prompt.

To delete all files without a file extension, enter "*.".

Example

IF you enter...	**THEN...**
DEL B:*.TMP	all files in drive B with the extension .TMP are deleted.

Remarks

To delete an actual directory (as opposed to all the files that a directory contains) you must use the RMDIR command.

ERASE is synonymous with DEL.

DIR

Lists details of the files in the current or a specified directory.

Classification

Internal

DIR [*pathname*] [**/P**] [**/W**]

Where

SYNTAX ELEMENT	**MEANING**
pathname	The specification of the file or directory to be listed (excluding the drive to list a file or directory in the default drive, excluding the directory path to list a file in the current directory).
/P	The directory display halts as soon as the screen becomes full. Press any key to resume the the listing.
/W	File and directory names only are displayed, five to a line across the screen.

Characteristics

If you do not specify a file name and extension, all files in the specified (or current) directory are listed.

You may use the wild cards * and ? in the file name and extension. If you omit either the name or the extension, the wild card * is assumed in its place.

To list a file that does not have an extension but exclude any that do in the pathname, enter the file name followed by a period (.).

DIR produces a display in which the size in bytes and date and time of creation or last modification appear alongside the file name(s).

Remarks

DIR does not display hidden files in a directory.

Examples

IF you enter...	THEN...
DIR B: (or DIR B:*.*)	all files in the drive B current directory are listed.
DIR .COM (or DIR *.COM)	all files in the default drive current directory having the extension COM are listed.
DIR AUTHORS (or DIR AUTHORS.*)	all files in the current directory on the default drive with the name AUTHORS and any extension are listed.

DISKCOMP

Compares the contents of two diskettes of the same type.

Classification

External

[*d*:][*path*] **DISKCOMP** [*drive1*: [*drive2*:]] [**/1**] [**/8**]

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where COMP is to be found.
path	Specifies the directory where COMP is to be found.
drive1	The drive to contain the first of the two diskettes to be compared.
drive2	The drive containing the diskette to be compared with the diskette in *drive1*.
/1	Only the first sides of the diskettes are compared.
/8	Only eight sectors per track are compared, even if the diskette in *drive1* has nine sectors per track.

Characteristics

The DISKCOMP command can only be used to compare diskettes. If you specify a hard disk drive an error message will be issued.

If neither drive is specified a single drive comparison is done on the default drive.

If only *drive1* is specified then *drive2* assumes the default drive.

The DISKCOMP command prompts you to insert the diskettes at the appropriate time then waits for you to strike any key before continuing.

The DISKCOMP command compares all tracks on the diskettes and indicates mismatched tracks by track and side number.

The DISKCOMP command determines the number of sides to be compared and the number of sectors per track from the first diskette. That is:

- If the first diskette is dual-sided and has nine sectors per track then a nine sectors per track comparison on both sides of the second diskette will be performed (unless **/1** and/or **/8** was specified). If the second diskette is single-sided or formatted eight sectors per track an error message will be displayed.
- If the first diskette is single-sided then only the first side of the second diskette will be compared regardless as to whether the second diskette is single or dual sided.
- If the first diskette is formatted eight sectors per track, then only eight sectors per track of the second diskette will be compared regardless as to whether the second diskette is formatted eight or nine sectors per track.

When all tracks have been compared the following prompt appears:

Compare more diskettes (Y/N)?

Press **Y** to perform another comparison using the same drives, or press **N** to exit the program.

Examples

IF you enter...	THEN...
DISKCOMP A: B:	the diskette in drive B is compared with the diskette in drive A.
DISKCOMP /1	a single-drive comparison is done using the default drive. Only the first sides of the diskettes are compared.
DISKCOMP B: /8	an eight sectors per track comparison is done between the diskettes in drive B and the default drive. If drive B is the default drive then a single-drive comparison is done.

Remarks

The DISKCOMP command compares entire diskettes. If you wish to compare only files you must use the COMP command.

The DISKCOMP command cannot be used to compare a diskette created using the COPY command with the original because the COPY command copies on a file-by-file (not track-by-track) basis.

On single-drive systems all prompts are for drive A, irrespective of the drive letters you entered in the command line.

DISKCOPY

Copies the contents of a diskette in one drive onto another diskette.

Classification

External, non-network

[*d*:][*path*] **DISKCOPY** [*sourcedrive*:] [*targetdrive*:] [/1]

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where DISKCOPY is to be found
path	Specifies the directory where DISKCOPY is to be found.
sourcedrive	The letter of the drive that contains the diskette to be copied.
targetdrive	The letter of the drive that contains the diskette to receive the copy.
/1	This switch indicates that only the first side of the diskette is copied.

Characteristics

The DISKCOPY command copies entire diskettes. Use COPY to copy files, or to copy to a different disk type than the source. For DISKCOPY, the diskettes must be of the same type, i.e. single, double, or high capacity. DISKCOPY automatically determines the number of sides to copy, based on the source drive and diskette. The target diskette is formatted or reformatted if necessary, during the copying. You can use the CHKDSK command to determine the capacity (and hence the format) of the source diskette.

You can specify the same drives or you may specify different drives. If the drives designated are the same, a single-drive copy operation is performed. You are prompted to insert the disks at the appropriate times. DISKCOPY waits for you to press any key before continuing. If you omit both parameters, a single-drive copy operation will be performed on the default drive.

If you omit the second option, the default drive will be used as the target drive.

After copying, DISKCOPY prompts:

Copy another disk (Y/N)?

If you press **Y**, the next copy is performed on the same drives that you originally specified, after you have been prompted to insert the proper diskettes. To end the COPY, press **N**.

Remarks

After an apparently successful DISKCOPY, you can carry out a DISKCOMP to compare the source and target diskettes.

If diskette errors are encountered during a DISKCOPY, you can run CHKDSK with the **/F** switch to try to correct errors on the source diskette. Use COPY *.* for each directory instead of DISKCOPY to copy the suspect diskette.

Do not use DISKCOPY when a directory on the source disk is JOINED to another drive.

ECHO

Turns the batch file echo feature off and on or outputs message to the standard output device.

Classification

Internal

ECHO [ON|OFF |*message*]

Characteristics

Normally, commands in a batch file are displayed (''echoed'') on the screen when they are interpreted by the command processor. ECHO OFF turns off this feature. ECHO ON turns the echo back on.

If ON or OFF are not specified, the current setting is displayed.

ECHO *message* outputs the message. Note that this message can be redirected using (< or > or > >)

Remarks

ECHO *message* can be used as a command outside of Batch files.

ERASE

Deletes the specified file(s)

See the ''DEL'' command.

EXE2BIN

Converts files from .EXE format to binary format.

Classification

External

[*d*:][*path*] **EXE2BIN** *pathname1* [*pathname2*]

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where EXE2BIN is to be found.
path	Specifies the directory where EXE2BIN is to be found.
pathname1	The file specification of the file to be converted (excluding the drive if it is in the default drive, excluding a directory path if it is in the current directory, excluding the extension to accept the default extension of .EXE).
pathname2	The file specification of the output file (excluding the drive and/or file name to accept the drive and/or file name in *pathname1* excluding a directory path to accept the current directory, excluding the extension to accept the default extension of .BIN).

Characteristics

The input file must be in valid .EXE format produced by the linker. The resident, or actual code and data part of the file must be less than 64K. There must be no STACK segment.

Two kinds of conversions are possible, depending on whether the initial CS:IP (Code Segment: Instruction Pointer) is specified in the .EXE file:

1. If CS:IP is not specified in the .EXE file, a pure binary conversion is assumed. If segment fixups are necessary (that is, the program contains instructions requiring segment relocation), you will be prompted for the fixup value. This value is the absolute segment at which the program is to be loaded. The resultant program will be usable only when loaded at the absolute memory address specified by a user application. The command processor will not be able to load the program.

2. If CS:IP is 0000:100H, it is assumed that the file will run as a .COM file with the location pointer set at 100H by the assembler statement ORG; the first 100H bytes of the file are deleted. No segment fixups are allowed, as .COM files must be segment relocatable; that is, they must assume the entry conditions explained in the MS-MACRO ASSEMBLER User Guide. Once the conversion is complete, you may rename the output file with a .COM extension. Then the command processor will be able to load and execute the program in the same way as the .COM programs supplied on your MS-DOS disk.

Remarks

If the input file does not meet one of the two requirements given above, the following message appears:

```
File cannot be converted
```

Note that to create a standard .COM file using the assembler you must set the location pointer at 100H using the ORG statement and use the END statement to set the first location as the start address. For example:

```
ORG       100H
START:
.
.
.
END       START
```

Do not have a .EXE file and a .COM file of the same name in the same directory, when you execute the file.

EXIT

Exits from a secondary command processor and returns to a parent program or command processor.

Classification

Internal

EXIT

Characteristics

This command can be used when you are running a program and have started a secondary MS-DOS command processor, then want to return to your program. For example, to look at a directory on drive B: while in the GW-BASIC environment, you must start the command processor by entering SHELL. The system prompt will appear:

```
A>
```

You can now enter the DIR B: command and MS-DOS will display the directory. When you enter EXIT, you return to the parent GW-BASIC environment.

FC

Compares the contents of two files.

Classification

External

[*d*:][*path*] **FC** [**/a**] [**/b**] [**/c**] [**/l**] [**/lb** *length*] [**/n**] [**/t**] [**/w**] [/#] *filename1 filename2*

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where FC is to be found.
path	Specifies the directory where FC is to be found.
filename1	The name of the first file to be compared.
filename2	The name of the second file to be compared.

Characteristics

The File Comparison utility, FC, compares the contents of two files. The difference between the two files can be output to the screen or to a third file. The files being compared may be either ASCII text or source files (files containing source statements of a programming language), or binary files (output from the assembler, the MS-LINK Linker utility, or a high-level language compiler).

The comparisons are either on a line-by-line or a byte-by-byte basis. The line-by-line comparison isolates blocks of lines that are different between the two files and prints those blocks of lines. The byte-by-byte comparison displays the bytes that are different between the two files.

The following table describes the switches available with the File Comparison utility. It is important to enter the switches in lower case, upper case switches are not recognised.

SWITCH	MEANING
/a	Abbreviates the output of an ASCII comparison. Instead of displaying all the lines that are different only the lines that begin each set of differences are displayed. The intermediate lines are represented by ellipses (...)
/b	A binary comparison of both files is performed. The two files are compared byte-to-byte, with no attempt to re-synchronize after a mismatch. The mismatches are printed as follows: --ADDRS----F1----F2 xxxxxxxx yy zz (where xxxxxxxx is the relative address of the pair of bytes from the beginning of the file). Addresses start at 00000000; yy and zz are the mismatched bytes from file1 and file2, respectively. If one of the files contains less data than the other, then a message is printed out. For example, if file F1 ends before file F2, then the utility displays:

SWITCH	MEANING
	fc: F2 longer than F1 This option is the default when you compare files with extensions of ''.EXE'', ''.COM'', ''.SYS'', ''.OBJ'', ''.LIB'' or ''.BIN''
/c	The utility ignores the case of letters. All letters in the files are considered upper case letters. For example, Much_MORE_data_IS_NOT_FOUND will match much_more_data_is_not_found This switch is used only in source comparisons.
/l	The utility compares the files in ASCII mode. It is the default when you compare files that do not have extensions of ''.EXE'', ''.COM'', ''.SYS'', ''.OBJ'', ''.LIB'' or ''.BIN''.
/lb *length*	Sets the Internal Line Buffer to *length* of lines. The default *length* of the internal buffer is 100 lines. Files that have more than *length* of consecutive differing lines will abort the comparisons.
/n	The line numbers are displayed on ASCII comparisons.
/t	Tabs are not expanded to spaces. The default is to treat tabs as spaces to 8 column positions.

SWITCH	MEANING
/w	The utility compresses whites (tabs and spaces) during the comparison. Thus, multiple contiguous whites in any line will be considered as a single white space. Note that although FC compresses whites, it does not ignore them. The two exceptions are beginning and ending whites in a line, which are ignored. For example (note that an underscore represents a white) ___More__data_to_be_found____ will match with More_data_to_be_found and with _______More_______data_to_be_____found_______ but will not match with ____Moredata_to_be_found This switch is used only in source comparisons.
/ #	Replace # with the number of lines required to match for the lines within the files to be considered as matching again, after a difference has been found. # can be any number from 1 to 9. If this switch is not specified, the number defaults to 2. This switch is used only in ASCII comparisons.

Note: The default setting for tabs is to convert them into spaces to 8-column positions.

The File Comparison utility reports the difference between the two files you specify by displaying the first file name, then the matching line before the differences followed by the lines that differ between the files, followed by the first line to match in both files. FC then displays the name of the second file followed by the matching line before the differences lines that are different, followed by the first line that matches. The default for the number of lines that must match before FC recognizes a match is 2. (If you want to change this default, specify the number of lines with the /# switch.) For example:

```
              ...
              ...
*****filename1
matching line before differences
difference
1st line to match file2 in file1 after difference

*****filename2
matching line before differences
difference
1st line to match file1 in file2 after differences
*****
              ...
              ...
```

If there are too many differences (involving too many lines), the program simply reports that the files are different and stops.

If no matches are found after the first difference is found, FC displays:

resynch failed. Files are too different

and returns to the MS-DOS default drive prompt.

The comparison report is sent to the screen unless you specify output redirection to a file.

FC uses a large amount of memory as buffer (storage) space to hold the source files. If source files are larger than available memory, FC compares only what can be loaded into the buffer space. If no lines match in those portions of the files that have been loaded into the buffer space, FC simply displays the message:

resynch failed. Files are too different

For binary files larger than available memory, FC compares both files completely, overlaying the portion in memory with the next portion from disk. All differences are output in the same manner as those files that fit completely in memory.

Examples

Assume these two ASCII files are on disk:

ALPHA.DOC	BETA.DOC
A	A
B	B
C	C
D	G
E	H
F	I
G	J
H	1
I	2
M	P
N	Q
O	R
P	S
Q	T
R	U
S	V
T	4
U	5
V	W
W	X
X	Y
Y	Z
Z	

The following examples show three possible ways of using FC to compare the contents of these two files:

IF you enter...	THEN...
FC ALPHA.DOC BETA.DOC	FC compares ALPHA.DOC with BETA.DOC and displays the differences on the screen. All the defaults remain intact. The output appears on the screen as follows (the Notes do not appear): ***** ALPHA.DOC NOTE: ALPHA file C contains CDEFG, D BETA contains CG E F G ***** BETA.DOC C G ***** ***** ALPHA.DOC NOTE: ALPHA file I contains IMNOP, M BETA contains N IJ12P O P ***** BETA.DOC I J 1 2 P ***** ***** ALPHA.DOC NOTE: ALPHA file V contains VW BETA W contains V45W ***** BETA.DOC V 4 5 W *****

IF you enter...	THEN...
	***** ALPHA.DOC
FC /4 ALPHA.DOC BETA.DOC >PRN	FC compares ALPHA.DOC with BETA.DOC. 4 lines have to be the same in the comparison, for FC to regard the lines within the file as matching again. The output is redirected to the line printer (PRN). ***** ALPHA.DOC C D E F G H I M N O P ***** BETA.DOC NOTE: P is the 1st C of a string of G 4 matches H I J 2 P ***** ***** ALPHA.DOC V W ***** BETA.DOC NOTE: W is the 1st V of a string of 4 4 matches

IF you enter...	THEN...
	5 W ***** ***** ALPHA.DOC
FC /b ALPHA.DOC BETA.DOC	the following binary comparison report appears: NOTE: The first field is the relative address of the part of bytes from the beginning of the file. The second field is the mismatching byte from ALPHA.DOC. The third field is the mismatching byte from BETA.DOC. 00000009: 44 47 0000000c: 45 48 0000000f: 46 49 00000012: 47 4a 00000015: 48 31 00000018: 49 32 0000001b: 4d 50 0000001e: 4e 51 00000021: 4f 52 00000024: 50 53 00000027: 51 54 0000002a: 52 55 0000002d: 53 56 00000030: 54 34 00000033: 55 35 00000036: 56 57 00000039: 57 58 0000003c: 58 59 0000003f: 59 5a fc: alpha.doc longer than beta.doc

FDISK

Sets up the MS-DOS partition for the hard disk.

Classification

External, Non-network

```
FDISK
```

Characteristics

The FDISK command allows you to set the MS-DOS partition on the hard disk. Refer to Chapter 2 for operational details.

FIND

Searches for a specific string of text in a file or files.

Classification

External

[*d*:][*path*] **FIND** [/V] [/C] [/N] "*string*"[*pathname*]...

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where FIND is to be found.
path	Specifies the directory where FIND is to be found.
string	A string of valid characters contained in quotes (").
pathname	The path of a file to be searched.

Characteristics

FIND displays all lines that contain the specified string from the file or files listed in the command line.

Put in two sets of quotes where the string itself contains quotes. E.g. FIND ''this is a quote ('''')'' finds the string; this is a quote ('').

You cannot use wild cards in your file specifications.

If no files are specified, FIND takes the standard input and displays all lines that contain the specified string.

Switches for FIND are:

SWITCH	MEANING
/V	Causes FIND to display all lines not containing the specified string.
/C	Causes FIND to print only the count of lines that contain a match in each of the files.
/N	Causes each line to be preceded by its relative line number in the file.

Example

IF you enter...	THEN...
FIND ''COLOR'' BOOK1 BOOK2	FIND displays each line in BOOK1 and BOOK2 (in that order) that contains the string COLOR.
DIR B:|FIND/V ''DAT''	FIND displays each file name on the disk in drive B that does not contain the string DAT. Note: when using piping in this manner your default drive diskette must not be write-protected.

Remarks

If you use the FIND command on BASIC text files, the text must have been saved in ASCII format.
If you use more than one switch with the FIND command, you will get the results summarized in the following table:

COMBINATION	RESULT
all three switches or **/V** and **/C**	The lines not containing the string are displayed.
/C and **/N**	The count of lines that contain the string is displayed.
/V and **/N**	The lines not containing the string are displayed, together with their line numbers.

FOR

Allows iterative execution of MS-DOS commands.

Classification

Internal

Interactive entry:

FOR %*char* **IN (** *item* **...) DO** [*command*] **%***char*

Batch entry:

FOR %%*char* **IN (** *item* **...) DO** [*command*] **%%***char*

Where

SYNTAX ELEMENT	MEANING
char	Any single character other than the digits 0-9.
item	A parameter valid for the command required (separated from another such item by a space).
pathname	The drive and/or directory where the command is to be found.
command	The command you wish to invoke.

Characteristics

Use the *%char* variable in a command line you enter interactively. Use the *%%char* variable in a command line within a batch file. The *%char* or *%%char* variable is assigned the value of each item listed in turn. The command specified is activated for each resulting parameter.

You may include the wild cards * and ? in an item.

Remember to separate each item with a space, and to surround the complete item list by brackets.

Examples

IF you enter in a batch file...	THEN...
FOR %%f IN (*.ASM) DO MASM %%f	all .ASM files are submitted to the assembler.
FOR %%f IN (report memo address) DO DEL %%f	the files report memo and address are deleted.

Remarks

A FOR command cannot call another FOR command directly. However a FOR command can call a secondary command processor, which in turn processes another FOR command (see details of COMMAND for a full explanation).

For example:

```
FOR %x IN (1 2) DO COMMAND /C FOR %y IN (%x) DO REM %y
```

produces the output:

```
A> COMMAND /C FOR %y IN (1) DO REM %y
A> REM 1
A>
A> COMMAND /C FOR %y IN (2) DO REM %y
A> REM 2
```

FORMAT

Formats a disk to receive MS-DOS files.

Classification

External, Non-network

[*d:*][*path*] **FORMAT** [*drive:*] **[/1] [/S] [/O] [/V] [/8] [/4]**

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where FORMAT is to be found.
path	Specifies the directory where FORMAT is to be found.
drive	The name of the drive that contains the disk. If the drive is not specified, the default drive is assumed.

Characteristics

You must run FORMAT on any new diskette you wish to use with MS-DOS. Any information already on the diskette is destroyed.

FORMAT places a bootstrap loader directory and file allocation table at the beginning of the diskette. It also checks for any faulty blocks on the diskette.

If you use the FORMAT command on a hard disk it will format the MS-DOS partition.

Switch options available with the FORMAT command have the following effect:

SWITCH	MEANING
/1	Formats a diskette single-sided. You would use this option for preparing 180 Kbyte diskettes on a double sided drive (or 160 Kbyte diskette if the /8 option is also specified).
/S	Copies the system files to the disk being formatted. These are: IO.SYS MSDOS.SYS COMMAND.COM IO.SYS and MSDOS.SYS are hidden files and will not show in a directory listing.
/O	Can only be used in conjunction with the /8 option to leave a place in the directory for the operating system of MS-DOS version 1.1. But the operating system is not placed on the disk. Note: This option causes the FORMAT program to take significantly longer.

SYNTAX ELEMENT	MEANING
/V	Allows you to enter a volume label. The FORMAT command issues a prompt that enables you to enter a unique volume label of up to 11 characters. This label will appear in subsequent directory listings.
/8	Formats diskettes 8 sectors per track instead of the default 9 sectors per track. Diskettes formatted in this manner are compatible with MS-DOS VER. 1.XX.
/4	Formats 48 tpi diskettes in High Capacity drives Note: Diskettes formatted with this switch can not reliably be used in Normal Capacity drives.

Remarks

Refer to the section on ''DISKS'', in Chapter 1 for a chart of 5 1/4'' Diskette Capacities. Refer to the section ''HOW TO FORMAT YOUR DISKETTES'' in Chapter 2 for a chart of ''Diskette Type Compatability in Different Capacity Drives''.

Unless you use a switch to specify otherwise, the default, format depends on the diskette type and the drive capacity.

Example

IF you enter...	THEN...
FORMAT B: /S	the diskette in drive B is formatted and operating system files are copied onto it.

Remarks

For diskette drives, FORMAT prompts you with a message such as:

```
Insert new diskette for drive B:
and strike ENTER when ready
```

When you have struck ENTER to continue MS-DOS formats the disk track-by-track. When it has finished you will receive a message such as:

```
Formatting ... Format Complete

362496 bytes total disk space
362496 bytes available on disk

   Format   another   (Y/N)?
```

Press **Y** to format another; **N** to return to MS-DOS.

GOTO

Jumps to a specified position in a batch file.

Classification

Internal

GOTO *label*

Where

SYNTAX ELEMENT	MEANING
label	A string of characters, the first eight of which are significant (there is no need for quotes around the string). In a batch file, when the GOTO command is executed, the next command executed is on the line following the label. Any line in a batch file can start with a : *label*. The contents of this line are not displayed by the MS-DOS batch processor. So preceding batch lines by : is useful for placing comments in a batch file. If the first eight characters of two labels are identical, GOTO that label will cause a jump to the first of the two labels. If no label is found the batch file terminates, with the message: **Label not found.**

Characteristics

To define a label in a batch file, precede a sequence of characters by a colon (:). Batch processing then ignores the line until it encounters the GOTO command with the label as parameter. It then jumps to the line below the one that contains the label.

Do not enter the colon when using the label as a parameter of GOTO.

Example

IF you execute a .BAT file containing...	THEN...
:foot REM looping... GOTO foot	an infinite number of ''looping...'' messages are produced.

GRAPHICS

Enables graphics currently displayed on the screen to be printed, on a compatible printer, along with any text when the **SHIFT PRT SC** keys are pressed.

Classification

External

[*d*:][*path*] **GRAPHICS** [*printer-type*] **[/B][/R][/D[/U | /H]]**

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where GRAPHICS is to be found
path	Specifies the directory where GRAPHICS is to be found.

The *printer-type* parameter can be:

PRINTER-TYPE	DESCRIPTION
graphics	Olivetti PR-15B or PR-17B or Industry Standard Graphics Printer or Epsom MS/FX 80
color1	Industry Standard Color Printer with black ribbon
color4	Industry Standard Color Printer with RGB (Red, Green, Blue and Black) ribbon 4 colors.
color8	Industry Standard Color Printer with CMY (Cyan, Magenta, Yellow and black) ribbon 8 colors.
jx80	Epsom JX-80 (color printer)
dm285 dm295	Olivetti PR-12 or PR-14 Color Printer

Remarks

If no *printer-type* parameter is given then graphics is assumed.

Switch options have the following effect:

SWITCHES	MEANING
/B	Causes the background color to be printed, otherwise the background is suppressed.
/R	Causes black on the screen to be printed black and white on the screen to be printed white. Without the switch the default is to print black as white and white as black.
/D	For a machine with an Enhanced Graphics Color Board (EGC) but no DEB INT 10 filter installed. Followed by one of the following this switch must be used.
/U	400 scanlines (Olivetti High Resolution)
/H	200 scanlines (Industry Standard Compatible)

Remarks

The first three switches are only of relevance if you have an Enhanced Graphics Color Board (EGC) installed. The last two switches affect the aspect of the printout.

Characteristics

This command must be entered to install graphics support necessary to print the screen in graphics modes. **SHIFT PRTSC** then invokes the printing function. Re-enter the GRAPHICS command with new parameters to reset the existing parameters; graphics support is not reinstalled, only the parameters are changed.

For details of printing with an Enhanced Graphics Color Board, see EGC Board User Guide (4021600 D)

The GRAPHICS command can only be used with printers that have graphics capabilities.

Remarks

Use INT5 to print the screen from a program.

Use the GRAPHICS command before entering GW-BASIC if you want to print graphics and text with the GW-BASIC LCOPY command. Text modes are printed in the upright position. Graphics Modes are rotated counterclockwise 90 degrees on the printout page, so the Visual Display Units upper right corner appears on the paper's upper left corner.

Warning

Do not turn the printer off while printing as this may cause unpredictable effects, and force you to reboot the operating system.

GRAFTABL

Loads the non-BIOS ASCII characters for graphics modes.

Classification

External

[*d*:][*path*] **GRAFTABL**

SYNTAX ELEMENT	MEANING
d	Specifies the drive where GRAFTABL is to be found
path	Specifies the directory where GRAFTABL is to be found.

Characteristics

After loading the character table, the following message is displayed.

GRAPHICS CHARACTERS LOADED

The routine is resident and occupies Random Access Memory space. If you try to load GRAFTABL again the following message is displayed.

GRAPHICS CHARACTERS ALREADY LOADED

As this command's exact operation is machine dependant, its actual working will be described in the Installation and Operations Guide of your particular personal computer.

Remarks

The Random Access Memory space occupied may only be reclaimed by rebooting the system.

GWBASIC

Enters the MS GW-BASIC Interpreter.

Classification

External

[*d*:][*path*] **GWBASIC**

SYNTAX ELEMENT	MEANING
d	Specifies the drive where GWBASIC is to be found.
path	Specifies the directory where GWBASIC is to be found.

Remarks

For more information on how to initialize GW-BASIC see the ''MS GW-BASIC Interpreter under MS-DOS User Guide''.

HEXDUMP

Displays the contents of a file, byte by byte, in hexadecimal.

Classification

External

[*d*:][*path*] **HEXDUMP** [*drive:*]*filename*

Where

SYNTAX ELEMENT	**MEANING**
d	Specifies the drive where HEXDUMP is to be found.
path	Specifies the directory where HEXDUMP is to be found.
drive	A drive letter, specifying the current directory of the specified drive.
filename	The file whose contents are to be displayed.

Characteristics

Each line of the display shows 16 bytes of information. At the left-hand end of the line the address of the first byte in the line is given. The hexadecimal value of each of the next 16 bytes then follows -two hex numbers per byte. The right-hand columns show the ASCII equivalent (if any) of the bytes displayed in that line.

Example

IF you enter...	THEN...
HEXDUMP B:ALPHABET	the content of the file named ''ALPHABET'' on the diskette in drive B are displayed in hexadecimal.

HEXDUMP produces a display similar to the following:

```
Dumping File: B:ALPHABET

0000: 20 20 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e "  abcdefghijklmn"
0010: 6f 70 71 72 73 74 75 76 77 78 79 7a 30 31 32 33 "opqrstuvwxyz0123"
0020: 34 35 36 37 38 39 2d 5e 40 5b 3b 3a 5d 2c 2e 2f "456789-^@[;:],./"
0030: 2e 7c 41 42 43 44 45 46 47 48 49 4a 4b 4c 4d 4e ".|ABCDEFGHIJKLMN"
0040: 4f 50 51 52 53 54 55 56 57 58 59 5a 5f 21 22 23 "OPQRSTUVWXYZ !"#"
0050: 24 25 26 27 28 29 3d 7e 60 7b 2b 2a 7d 3c 3e 3f "$%&'()=`~{+*}<>?"
0060: 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 "                "
0070: 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 0d "               ."
0080: 0a                                              "."

HEXDUMP Complete
```

Fig. 6-2 HEXDUMP Display

IF

Causes conditional execution of a command in a batch file.

Classification

Internal

IF [NOT] *condition command*

Where

SYNTAX ELEMENT	MEANING
condition	One of the three valid conditions listed below.
command	The command you wish to conditionally execute. If the command is external, it may optionally be preceded by: the drive where it is to be found; the path leading to the directory where it is to be found.

Characteristics

The specified command is only executed if the condition is true. If it is false the command is ignored. Valid conditions are as follows:

CONDITION	MEANING
EXIST [*d*:][*path*] *filename*	The command is executed only if the specified file exists. on drive *d*:, and in the directory to which the path leads. The default drive is searched if *d*: is not specified. The current directory is searched if *path* is not specified.
string1 = =string2	The command is executed only if the two strings are identical after parameter substitution. The case of the characters in *string1* and *string2* is significant.
ERRORLEVEL *number*	The command is executed only if the previous program executed had an exit code of the specified *number*, or higher.

Examples

IF you enter...	THEN...

IF NOT EXIST \SPECIAL\MARKER C:\BIN\
CREATE C:\SPECIAL \MARKER
	if the file MARKER does not exist the user program CREATE is run to create it.

IF %1 = = OLIVETTI ECHO PARAMETER 1 IS OLIVETTI	the computer displays the message only if parameter 1 after substitution is the string "OLIVETTI".
IF NOT ERRORLEVEL 3 LINK	if the error level is under 3 the linker runs.

JOIN

Joins a disk drive to an empty directory on another drive to produce a single directory structure.

Classification

External, Non-network

Syntax 1 - To join

[*d*:][*path*] **JOIN** *connected-drive splice-drive:\splice-directory*

or

Syntax 2 - To deassign a join

JOIN [*connected-drive*:**/D**]

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where JOIN is to be found.
path	Specifies the directory where JOIN is to be found
connected-drive	The drive which is to be connected to another drive.
splice-drive	The drive to which reference is to be made.
splice-directory	The connected drive's directory structure is spliced to this directory.
/D	This switch indicates that the connected drive is to be unspliced.

Characteristics

The JOIN command removes the distinction that physical drives are separately addressable by drive letter. You can refer to all the directories on the joined drives as a single tree structure on one logical drive. After the JOIN command is executed the *connected-drive* becomes invalid.

Remarks

You can only join a *connected-drive* to a *splice-directory* which is a sub-directory of the root of the *splice-drive*. If the *splice-directory* does not exist JOIN will create it. If the *splice-directory* does exist it must be empty of files and sub-directories. Do not JOIN a drive, if the drive being JOINED is part of a substitution (SUBST) or assigned (ASSIGN).

Example: To join drive A: (the *connected–drive)* to drive C: (the *splice-drive)*.
Before the join the ''C: Drive Directory Structure'' is as shown in the following figure:

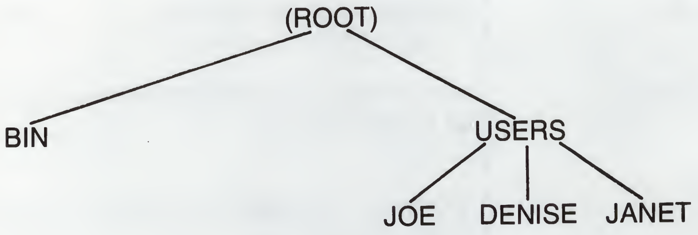


Fig. 6-3 C: Drive Directory Structure

Before the join the ''A: Drive Directory Structure'' is shown in the following figure:

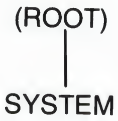


Fig. 6-4 A: Drive Directory Structure

after the command

```
JOIN A: C:\DRIVE-A
```

The "Spliced C: Drive Directory Structure appears to be:

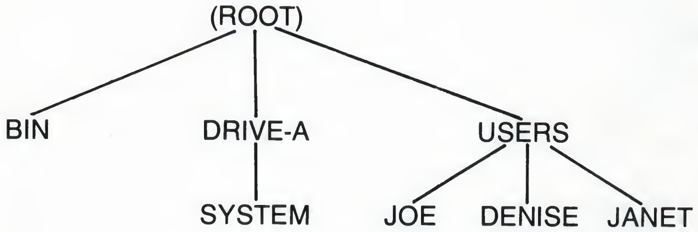


Fig. 6-5 Spliced C: Drive Directory Structure

Notice that in the ''Spliced C: Drive Directory Structure'' that the sub-directory DRIVE-A has been created; the sub-directory SYSTEM of A: has been spliced into C: and the root directory of drive A: has been been replaced by the sub-directories DRIVE-A. The whole directory structure of the connected drive is always joined. Drive A: is inaccessible while JOINED. If you try to refer to it the error message is output:

```
Invalid drive specification
```

To find out which drives are joined enter **without parameters:**

```
JOIN
```

in the above example the following message is output:

```
A: => C:\DRIVE-A
```

To deassign the join in the example enter:

```
JOIN A: /D
```

Note: The current directory of the *splice-drive* should always be \(root) when JOIN commands are executed. When joining drives, the *connected-drive* should not be the default drive.

Warning

The following commands should not be used while drives are JOINED:

- BACKUP
- DISKCOMP
- DISKCOPY
- FORMAT
- RESTORE

LABEL

Creates, changes or deletes a disk volume label.

Classification

External, Non-network

[*d*:][*path*] **LABEL** [*drive:*][*volume-label*]

SYNTAX ELEMENT	MEANING
d	Specifies the drive where LABEL is to be found.
path	Specifies the directory where LABEL is to be found.
drive	The drive containing the disk you wish to LABEL.
volume-label	The name you wish to give the disk. Refer to the section ''PARAMETERS'' in Chapter 5 for the syntax details of *volume label*. From 1 to 11 characters including spaces are allowed.

Characteristics

If you do not specify a volume label, LABEL prompts

Volume in drive X is xxxxxxxxxxx
Volume label (11 characters, ENTER for none)?

Type the volume label that you want and press the **ENTER** key. If you want to delete the volume label, just press the **ENTER** key. LABEL prompts with the message:

Delete current volume label (Y/N)?

If you press **Y**, Label deletes the volume label on the disk, otherwise the volume label remains unchanged.

MKDIR

Creates a new directory.

Classification

Internal

Syntax 1

MKDIR *path*

Syntax 2

MD *path*

Where

SYNTAX ELEMENT	MEANING
path	The path of the directory you want to create. With or without a drive specifier.

Characteristics

Use MKDIR to create or add to a hierarchical directory structure on the disk in the default or specified drive.

You may enter either MKDIR or MD to invoke this command.

Example

IF you enter...	THEN...
MKDIR \USER	the subdirectory USER is created beneath the root directory in the default drive.

MODE

This command enables you to:

- Set the protocol for an RS-232-C or current loop port.
- Set the monitor mode.

- Set the printing characteristics or redirect printer output to the communications port.

Classification

External

MODE COM:

Sets the protocol for an RS-232-C or current loop port.

[*d*:][*path*] **MODE COM***n*:*baud*[, *parity*[, *databits*[, *stopbits*[, **P**]]]]

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where MODE is to be found.
path	Specifies the directory where MODE is to be found.
n	One of the following values: 1 the built-in RS-232-C port 2 the (optional) second RS-232-C or current loop port.

▶

SYNTAX ELEMENT	MEANING
baud	The baud rate. This must be one of the following 110, 150, 300, 600, 1200, 2400, 4800 or 9600. Only the first two digits need be specified.
parity	One of: - **E** (even) - **O** (odd) - **N** (none) The default is **E** (even).
databits	The number of data bits. This must be either 7 or 8. The default is 7.
stopbits	Either 1 or 2. If the baud rate is specified as 110, then the default is 2, otherwise the default is 1.
P	Continous retry on time-out errors.

Characteristics

The baud rate must be specified. All other parameters are optional and will take default values if omitted.

Parameters must be separated by commas.

Use the **P** switch with a serial interface printer. Retry loops resulting from a repeated time-out condition can be broken by typing **CTRL BREAK.**

Examples

IF you enter...	THEN...
MODE COM1:11,O,8,1	the baud rate is set to 110, odd parity is specified, and the data bits and stop bits are specified as 8 and 1, respectively.
MODE COM1:96	the baud rate is set to 9600. All other parameters take default values.

MODE n

Sets the monitor mode.

Syntax 1

[*d*:][*path*] **MODE** *n*

Syntax 2

[*d*:][*path*] **MODE** [*n*],*m*[, **T**]

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where MODE is to be found.
path	Specifies the directory where MODE is to be found.
n	An argument that may take one of the following values: 40 The width of the display is set to 40 characters per line (color monitor only). 80 The width of the display is set to 80 characters per line (color monitor only). BW40 Switches the active display controller to that of the color display, sets the display mode to black and white and the display width to 40 characters per line. BW80 Switches the active display controller to that of the color display, sets the display mode to black and white and the display width to 80 characters per line. CO40 Switches the active display controller to that of the color display, sets the display mode to color and the display width to 40 characters per line. CO80 Switches the active display controller to that of the color display, sets the display mode to color and the display width to 80 characters per line.
m	Is **R** for right shift and **L** for left shift.
T	Displays a test pattern at the top of the screen.

Characteristics

Use the *m* parameter to shift the display one character (with 40 columns) or two characters (with 80 columns) to the left or right. If you specify the **T** switch, MODE displays a series of numbers at the top of the screen, and prompts you:

Do you see the leftmost 0 (Y/N) ?

if you have entered **R**, or

Do you see the rightmost 9 (Y/N) ?

if you have entered **L**.

In response to either prompt press **N** to shift the display and redisplay the prompt, press **Y** to return to MS-DOS.

Remember, if you wish to omit the *n* parameter, to precede the *m* parameter by a comma.

MODE LPT

Sets the mode of operation for a compatible printer, or redirects the output for any printer to a communications port.

Syntax 1

[*d*:][*path*] **MODE LPT**#: [*chars*][,*spacing*] [,**P**]

Syntax 2

[*d*:][*path*] **MODE LPT#:=COM***n*

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where MODE is to be found.
path	Specifies the directory where MODE is to be found.
#	The printer number (1, 2 or 3).
chars	The number of characters per line.
spacing	Vertical spacing in lines per inch. Its value must be either 6 or 8.
n	The number of the communications port to which printer output is to be redirected.
P	Override time-out error conditions using the port

Example

IF you enter...	THEN...
MODE LPT1: 132,8	The mode of operation of printer 1 is set to 132 characters per line and 8 lines per inch
MODE LPT1: = COM1	Output that would normally be sent to the printer is redirected to the first RS-232-C port.

MORE

Sends output to the terminal one screen at a time.

Classification

External

```
[d:][path] MORE
```

Characteristics

Either redirect input through MORE or use MORE as a filter added at the end of a command line. Screen output is displayed one screen at a time, instead of scrolling through its entire contents. At the bottom of each screenful, the prompt **---MORE---**is displayed. Press any typing key to display the next screenful.

Example

IF you enter...	THEN...
TYPE MYFILE|MORE or MORE <MYFILE	The file MYFILE on the default drive is displayed one screen at a time. Note: The diskette cannot be write-protected.

PATH

Sets a command search path in the environment.

Classification

Internal

PATH = [;|[*pathname*[;*pathname*]...]]

Where

SYNTAX ELEMENT	MEANING
pathname	The path of a directory you wish MS-DOS to search, including optionally a drive letter. Do not include a filename.

Characteristics

PATH tells MS-DOS which directories, and in what order, to look for external commands after it has searched your current directory.

You can specify a single directory path or a number of paths each separated by a semicolon (;). The default is no search path; in this case MS-DOS only searches your current directory.

If you enter **PATH** with no parameter, MS-DOS displays the current search path. If you enter **PATH;** any previously established path is cancelled and only your current directory is searched.

You only need to set the search path once in any terminal session.

Path only finds executable files: e.g. .COM, .EXE, .BAT files. Path ignores files with any other extension. The paths are searched in the order specified, so place the most frequently accessed directories first.

Example

IF you enter...	THEN...
PATH C:\BIN\USER;C:\BIN\DEV	MS-DOS searches first your current directory, then \BIN\USER, and finally \BIN\DEV on the C: drive.

Remarks

Non-existent directories specified in the PATH variable in the environment are ignored.

PAUSE

Suspends execution of the batch file in which it is contained.

Classification

Internal

PAUSE [*comment*]

Where

SYNTAX ELEMENT	MEANING
comment	A string of up to 121 characters.

Characteristics

When PAUSE is encountered during the execution of a batch file, any comment you have entered shows on the monitor followed by this prompt:

Strike a key when ready...

At this point the batch file is suspended, allowing you to change disks or perform any other necessary action.

To resume batch execution press any key with the exception of **CTRL C**.

Press **CTRL C** to cancel processing of the batch file. The following prompt appears:

> **Abort batch job (Y/N)?**

Press **Y** to cancel the batch operation and return to the MS-DOS prompt. Press **N** to return to the previous prompt.

Example

IF you enter...	**When the batch file runs...**
PAUSE insert target disk in drive B:	the batch job is suspended and the appropriate prompt is displayed.

PRINT

Queues text files for background printing, while other MS-DOS commands are obeyed.

Classification

External, Non-network

Syntax 1

The first time PRINT is called

[*d:*][*path*] **PRINT** [**/D**:*device*][**/B**:*buffsize*][**/U**:*busyticks*][**/M**:*maxticks*][**/S**:*timeslice*][**/Q**:*queuesize*] [[*pathname*]...]

Syntax 2

Subsequent calls to PRINT

[*d*:][*path*] **PRINT** [**/C**|**/P**] [[*pathname*][**/C**|**/P**]...]

Syntax 3

Subsequent call to terminate PRINT

[*d*:][*path*] **PRINT /T**

Where

SYNTAX ELEMENT	**MEANING**
d	Specifies the drive where PRINT is to be found.
path	Specifies the directory where PRINT is to be found.
filespec	The file specification of a file to be printed.
/D:*device*	Use to specify the print device. If not used PRINT will ask for a print device.
/B:*buffsize*	Use to set the internal print buffer size in bytes. The normal size is 512 bytes. Increasing the size may increase performance.

▶

SYNTAX ELEMENT	MEANING
/**U**:*busyticks*	Specifies the number of MS-DOS clock ticks that PRINT will wait if the printer is busy. Otherwise PRINT gives up its timeslice. The default is 1 tick.
/**M**:*maxticks*	Specifies how many MS-DOS clock ticks PRINT can have to print a file. *maxticks* can be from 1 to 255 clock ticks (the default is 2).
/**S**:*timeslice*	Specifies the time slice value. *timeslice* can be from 1 to 255 (the default is 8). The lower the value the higher the priority of the print queue.
/**Q**:*queuesize*	Specifies the number of files allowed in the print queue. *queuesize* can be from 1 to 32 (the default is 10).

Characteristics

You may use global and wildcard characters.

If you enter other commands during printing, printing is suspended until the command is completed.

When you run PRINT for the first time in a terminal session, you are prompted as follows:

Name of list device [PRN:]

Type the name of a valid line printer, or simply press **ENTER** to accept the default line printer.

The following switches are possible with this command:

SWITCH	MEANING
/T	TERMINATE: this switch cancels all files in the print queue (those waiting to be printed). A message to this effect will be printed.
/C	CANCEL: This switch turns on cancel mode. The preceding *pathname* and all following *pathname* will be suspended in the print queue until **/P** switch is encountered on the command line.
/P	PRINT: This switch turns on print mode. The preceding *pathname* and all following *pathname* will be added to the print queue until a **/C** switch is encountered on the command line.

Print with no parameters displays the contents of the print queue on your screen without affecting the queue.

Examples

IF you enter...	THEN...
PRINT /T	the print queue is emptied.
PRINT*.ASM	All the *.ASM files are queued to the printer.
PRINT TEMP1/C TEMP2 TEMP3	the three files indicated are removed from the print queue.

IF you enter...	THEN...
PRINT TEMP1/C TEMP2/P	TEMP1 is removed from the queue, whereas TEMP2 is added.

Warning

When PRINT is active do not print screen (**SHIFT PRT SCR**) or turn the printer echo on (**CTRL PRT SCR**).

PROMPT

Sets the MS-DOS command prompt.

Classification

Internal

PROMPT [{*meta-character*|*character*}...]

Where

SYNTAX ELEMENT	MEANING
meta-character	A special character you wish to use to create the MS-DOS prompt, preceded by a $ sign.
character	A character you wish to appear in your prompt, but this cannot be a $ or any of the characters described in the "MEANING" column of the following table.

Characteristics

If no argument is entered, the prompt will be set to the default prompt, which is the default drive designation plus the > symbol. You can set the prompt to something different such as the current time, by using the meta-characters indicated below.

The following meta-characters can be used in the prompt command to specify special prompts. They must all be preceded by a dollar sign ($) in the prompt command:

SPECIAL CHARACTER	MEANING
$	The '$' character.
t	The time.
d	The date.

▶

SPECIAL CHARACTER	MEANING
p	The default drive and the path to the current directory.
v	The version number.
n	The default drive.
g	The '>' character.
l	The '<' character.
b	The '/' character.
_	A carriage return-linefeed sequence.
s	A space (leading only).
h	A backspace.
e	ASCII ESCape (Hexadecimal 1B).
q	The '=' character.

Examples

IF you enter...	THEN...
PROMPT HELLO	the prompt becomes HELLO
PROMPT pg	the default drive and the current directory is established as a prompt, E.g. C:BIN>
PROMPT HELLO$g	the prompt becomes HELLO>

RECOVER

Recovers a file or an entire disk containing faulty sectors.

Classification

External, Non-network

[*d*:][*path*] **RECOVER** [*pathname*]

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where RECOVER is to be found.
path	Specifies the directory where RECOVER is to be found.
pathname	A drive or a directory or a file name containing a faulty sector.

Characteristics

Specify a file name to recover a particular file. RECOVER reads the file sector by sector and marks any faulty sectors it finds. MS-DOS will no longer allocate data to such sectors.

Specify a drive name to recover a complete disk. RECOVER reads the contents of the disk sector by sector and marks any faulty sectors it finds.

When you run RECOVER on a disk any directory tree that may be present is destroyed. Files are placed in the root directory and renamed FILE0001.REC, FILE0002.REC and so on. If there is not enough space in your root directory for information on all the files on the disk, the following message is displayed:

```
Warning - directory full
```

You can neither run RECOVER on a directory nor use a list of files or wild cards.

REM

Displays a remark during the execution of a batch file.

Classification

Internal

REM [*remark*]

Where

SYNTAX ELEMENT	MEANING
remark	A string of up to 123 characters.

Characteristics

A remark inserted in a batch file shows on the screen as soon as it is encountered during batch execution. ECHO OFF prevents display of *remark*.

REN

Renames files.

Classification

Internal

REN[AME]*pathname filename*

Where

SYNTAX ELEMENT	MEANING
pathname	The path of the file to be renamed (excluding the drive only if the file is on the default drive).
filename	The new name including any extension you wish to give the file.

Characteristics

REN changes the name and extension of the file specified in the first parameter to those given in the second parameter. REN cannot be used to move a file from one drive or directory to another; therefore the second parameter must only be a filename.

The wildcard filenames, using * and ?, may be used in either parameter. If wild cards appear in the second parameter, the corresponding characters in the first parameter remain unchanged.

Examples

IF you enter...	THEN...
REN B:PRESENT PAST	the file PRESENT in the current directory in drive B is renamed PAST.
RENAME *.JON *.?A?	any file in the current directory in the default drive with the extension JON has its extension changed to JAN.

Remarks

An attempt to give a file a name already in the file directory results in the following message:

Duplicate file name or file not found

RESTORE

Restores a number of files from back-up disks. The backup disks must have been created using the BACKUP command.

Classification

External

[*d:*][*path*] **RESTORE** *source-drive:* [*target-drive:*] [*pathname*] [**/S**] [**/P**] [**/B:***date*] [**/A:***date*] [**/E:***time*] [**/L:** *time*] [**/M**] [**/N**]

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where RESTORE is to be found.
path	Specifies the directory where RESTORE is to be found.
source-drive	The drive containing the disk with the backup information to be restored. Typically a floppy disk drive.
target-drive	The drive containing the disk to which the backup information is to be restored. Typically a hard disk drive.
pathname	The file(s) you wish to restore. If you enter only the hard disk specifier, then all files backed up from the current directory are restored. If you specify a path terminating in a directory name then all files backed-up from that directory will be restored to it. If the path terminates in a file name (or a group of file names specified using wild card characters) then only the specified file(s) will be restored.
/S	Files in all subdirectories, as well as those in the specified (or current) directory will be restored. This includes all levels of subdirectory below the specified (or current) directory.

▶

SYNTAX ELEMENT	MEANING
/P	You will be prompted before restoring files that have been modified since the backup was made, or those files that are read only. This switch is recommended when restoring files backed up from MS-DOS 2.11 disks.
/B:*date*	This switch indicates that only those files which were modified on or before the given *date* should be restored.
/A:*date*	This switch indicates that only those files that were last modified on or after the given *date* should be restored.
/E:*time*	This switch indicates that only those files which were modified at or earlier than the given *time* should be restored.
/L:*time*	This switch indicates that only those files that were last modified at or later than the given *time* should be restored.
/M	This switch indicates that only those files that have been modified since the last backup should be restored.
/N	This switch indicates that only those files that no longer exist on the restore-drive should be restored.

Characteristics

Once you have entered the command line you are prompted to insert the backup diskette. It is up to you to ensure that you insert the diskette(s) containing the file(s) you wish to restore in the correct order. If you are unsure as to which diskette(s) contain the files you require, start with the first backup diskette then insert each backup diskette in turn in the order in which they were made. The RESTORE command will prompt you to insert the next diskette.

The RESTORE command sets the exit code as follows:

- 0 Normal completion
- 1 The specified file(s) was not found
- 3 Command execution terminated by the user
- 4 Command execution terminated due to error

The exit codes can be used by the batch processing IF command.

Examples

IF you enter...	THEN...
RESTORE A: C:\ /S	all files on a series of backup diskettes in drive A are restored to the hard disk drive C in the same directory structure.
RESTORE A: C:*.COM	all files on the backup diskette having the file name extension .COM that were backed up from the current directory are restored into the current directory on the hard disk drive C.

RMDIR

Removes an empty sub-directory

Classification

Internal

Syntax 1

RMDIR *pathname*

Syntax 2

RD *pathname*

Where

SYNTAX ELEMENT	MEANING
pathname	The path of the directory you wish to remove.

Characteristics

RMDIR removes a directory that is empty apart from the . and .. special files. If the directory contains subdirectories or files, these must first be removed by means of the RMDIR or DEL command respectively.

You may either enter RMDIR or RD to invoke this command.

Example

IF you enter...	THEN...
RMDIR C:\BIN\USER\JOE	the specified empty directory is removed, on the C: drive.

SELECT

Copies your MS-DOS system diskette to create a working copy for your selected country and keyboard.

Classification

External

[*d*:][*path*] **SELECT** *country-code keyboard-code*

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where SELECT is to be found.
path	Specifies the directory where SELECT is to be found.
country-code	A three digit number which is the telephonic international country code. Refer to the Command COUNTRY in Appendix C for details.
keyboard-code	A two character alpha code indicating your national keyboard. Refer to your Installation and Operations Guide for details.

Characteristics

Your source diskettes should be in drive A: and your target diskette in drive B:. For safety your source diskette should be write protected and your target diskette should be an unprotected scratch diskette. SELECT uses DISKCOPY to copy the system diskette to drive B:. It also creates on B: a CONFIG.SYS with

COUNTRY = *country-code*

and an AUTOEXEC.BAT with the command

KEYB <*keyboard-code*>

(but not for a United States keyboard).

Remarks

The correct keyboard driver must be present on your working system diskette before you use it to bootstrap the system. See your Installation and Operations Guide for more details on setting up your Systems Diskette keyboard and screen support. Refer also to Chapter 2 in this MS-DOS User Guide.

WARNING

This command inadvertently used can overwrite directories and files on the diskette in the B: drive.

SET

Assigns one string value in the environment to another key string; for use in programs or batch files.

Classification

Internal

SET [*key*=*value*]

Where

SYNTAX ELEMENT	MEANING
key	The key string you wish to assign a value to.
value	The string you wish to assign to the key string.

Characteristics

Use SET to assign a value to a standard parameter included in an application program. The value remains operative during a working session until another SET command is issued.

When the SET key command is executed, with a key assignment, it inserts the entire string into a part of memory reserved for ''environment'' strings. If the name already exists in the environment, it is replaced by the new string. If you type the Set command with only the first string, the associated string name is removed from the environment. If you type Set with no options, MS-DOS displays the current environment settings.

A program can get a listing of all environment values that have been set by examining its environment. Environments are passed in the Program Segment Prefix. Refer to the MS-DOS Ver. 3.1 System Programmers Guide for more information.

You can also use the SET command with batch files. Instead of passing string values to a batch file by means of replaceable parameters (see Chapter 5) in the command line, you can use SET to assign string values to string keys. Within the batch file the form of the *key* must be as follows:

%*key*%

That is, the string must be preceded and followed by a percentage sign (whereas replaceable parameters are only preceded by a percentage sign).

Example

IF you enter...	THEN...
SET TTY = VT52	the TTY value is set to VT52, in all batch files on execution %TTY% is replaced by VT52.

Remarks

The case of the key is converted to uppercase. The case of the value is left in the case input. Be careful because programs usually require value in uppercase, these values must be input in uppercase.

For example:

IF you enter...	THEN...
SET tty = vt52	in the environment tty -> TTY

SET	COMSPEC = A:\COMMAND.COM
	PATH =
	TTY = vt52

If the program expects VT52 in uppercase vt52 in lowercase will not be recognised.

SHARE

Installs MS-NET file sharing and locking.

Classification

External

[*d*:][*path*] **SHARE** [**/F:** *memory-space*] [**/L:** *locks*]

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where SHARE is to be found.
path	Specifies the directory where SHARE is to be found.
/**F**:*memory-space*	When this switch is used, the *memory-space* parameter indicates how much *memory space* is reserved for recording file sharing information. Each open file should be allocated space for its pathname plus eleven bytes; so the space allocated should be between 32 bytes and 74 bytes per file. 74 bytes will allow for a full-length pathname of 63 characters. The default *memory-space* for filesharing information is 2048 bytes.
/**L**:*locks*	When this switch is used, memory space is allocated for the maximum number of locks it is possible to apply to a file. The default value for the number of *locks* it is possible to apply is 20.

Characteristics

This command should only be used in conjunction with MS-NET. If used it should be included in the system disk's AUTOEXEC.BAT file. Once the command has been called the support utility becomes resident. It takes about 5k bytes of memory (with the default switch settings).

SHIFT

Allows access to more than 9 replaceable parameters in batch processing.

Classification

Internal

SHIFT

Characteristics

Usually, batch files are limited to handling 10 parameters, %0 through %9. To allow access to more than nine (replaceable) parameters, use SHIFT to alter the numbering of your command line parameters. For example:

caling a batch file SUPER.BAT with the 12 parameters:

```
                SUPER p1,p2,p3,p4,p5,p6,p7,p8,p9,p10,p11,p12
initially:      %0 = SUPER
                %1 = p1
                    .
                    .
                    .
                %9 = p9
```

to access the other parameters SHIFT is used within the batch file one SHIFT will result in:

```
%0 = p1
%9 = p10
```

Successive SHIFT calls will result in:

```
%9 = p11
```

and another SHIFT call will result in:

```
%9 = p12
```

so finally

```
%0 = p3
%1 = p4
%2 = p5
%3 = p6
%4 = p7
%5 = p8
%6 = p9
%7 = p10
%8 = p11
%9 = p12
```

So if you have entered more than nine parameters on the command line, those that appear after the ninth (%9) will be shifted one at a time into %9 by successive shifts. You can then refer to these parameters in your batch file.

SORT

Sorts data alphabetically, in forward or reverse order.

Classification

External

[*d*:][*path*] **SORT** [*pathname*] **[/R]** [/ + *number*]

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where SORT is to be found.
path	Specifies the directory where SORT is to be found.
pathname	The path of the file to be sorted.
/R	This switch indicates a reverse ASCII sort.
/ + *number*	The column number from which to begin the sort.

Characteristics

Sort is a filter which works on text lines, the case of the characters is ignored.

Sort takes place in ASCII order unless you specify the **/R** switch, in which case a sort in reverse ASCII sequence is done. It starts with the first column of input unless you specify otherwise using the / + *number* switch.

If you do not specify a file, SORT takes the standard input and outputs to the screen, unless you specify otherwise with the redirection symbols "> and <" or the pipe symbol "|".

Examples

IF you enter...	THEN...
SORT /R <UNSORT.TXT >SORT.TXT	the file UNSORT.TXT is sorted in reverse order and the result placed in the file SORT.TXT.
DIR|SORT/+14 |MORE	the directory listing produced by the DIR command is sorted starting with the fourteenth column (the column that contains the file size), and output a screen at a time.

SUBST

Substitute a dummy drive specifier for a pathname.

Classification

External, Non-network

Syntax 1

[*d*:][*path*] **SUBST** *dummy-drive: pathname*

or

Syntax 2

SUBST [*dummy-drive:*] [**/D**]

Where

SYNTAX ELEMENT	**MEANING**
d	Specifies the drive where SUBST is to be found.
path	Specifies the directory where SUBST is to be found.
dummy-drive	The dummy drive that is to be used to refer to pathname.
pathname	The drive and/or directory path to which the dummy drive refers.
/D	A switch which indicates that the specified dummy drive substitution should be deleted.

Characteristics

Type SUBST with no parameters to display the current substitution, for example:

M => C:\USR\MIKE

The *dummy-drive* must be within the range of drives recognized by the system. If you use real drive letters for the *dummy-drive*, you will not be able to use the real drive. Do not use the default drive as a *dummy-drive*. It is recommended that you do not use real drives. Increase the availability of drive letters by setting in CONFIG.SYS *LASTDRIVE = dummy-drive* or greater (see appendix E).

If you have a configuration of PC with a hard disk (C:) and two floppy disk drives (A: and B:), and are using a dummy drive = M. In CONFIG.SYS set LASTDRIVE = M. Call the command:

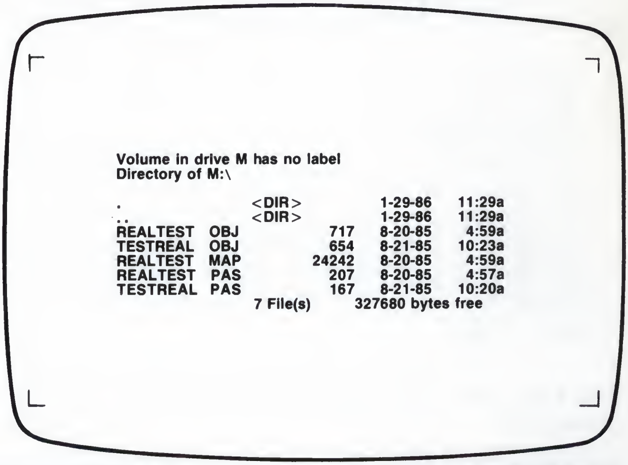
```
SUBST M: C:\USR\MIKE
```

where MIKE is a directory. You would now be able to refer to the directory by using the alias M: instead of the path C:\USR\MIKE.

After substitution the command:

```
    DIR M:
```

could product the following display:



```
Volume in drive M has no label
Directory of M:\

.                 <DIR>            1-29-86    11:29a
..                <DIR>            1-29-86    11:29a
REALTEST   OBJ               717   8-20-85     4:59a
TESTREAL   OBJ               654   8-21-85    10:23a
REALTEST   MAP             24242   8-20-85     4:59a
REALTEST   PAS               207   8-20-85     4:57a
TESTREAL   PAS               167   8-21-85    10:20a
                  7 File(s)       327680 bytes free
```

Fig. 6-1 Substituted Directory Display

Substituting is particularly useful for programs that do not recognise paths, or you can use a letter as shorthand for a long path.

To delete a substitution use Syntax 2.

In the above example:

```
SUBST M: /D
```

will delete the substitution.

Remarks

Never use the SUBST command and then use the following commands on the dummy drive or unpredictable results and/or error messages will occur.

ASSIGN
BACKUP
DISKCOMP
DISKCOPY
FDISK
FORMAT
JOIN
LABEL
PRINT
RESTORE

Pay attention to the substitutions in effect when using the following commands.

CHDIR
MKDIR
RMDIR
PATH

SYS

Updates the specified disk with the system files IO.SYS and MSDOS.SYS, which come from the default drive.

Classification

External, Non-network

[*d*:][*path*] **SYS** *drive:*

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where SYS is to be found.
path	Specifies the directory where SYS is to be found.
drive	The drive that contains the target disk.

Characteristics

The source disk is in the default drive.

The target disk must either be formatted, but without files in the root directory, or have been formatted with the **/S** parameter to contain previous versions of the system files. If this is not the case one of the following messages appears:

SYS cannot install MS-DOS on this disk

or

Not enough room for MS-DOS on this disk

Remarks

The file COMMAND.COM is not transferred.

IO.SYS and MSDOS.SYS are hidden files and will not appear in any directory listing.

TIME

Displays or set the system time.

Classification

Internal

TIME [*hh*[:*mm*]]

Where

SYNTAX ELEMENT	MEANING
hh	Hours (0-24).
mm	Minutes (0-59).

Characteristics

If you enter one or more value(s) and then press **ENTER**, TIME sets the remaining value(s) to zero. For example, if you enter ''8'' the time is set to 8:00:00.00

If you leave out the complete parameter, TIME prompts you as in the following example:

```
Current time is 16:36:00.00
Enter new time:
```

You can then enter a new time in the correct format. To accept the current time simply press **ENTER** .

Example

IF you enter...	THEN...
TIME 8:30	half past eight in the morning is set as the current time.

Remarks

Note that the format of the time output varies depending on the COUNTRY configuration in CONFIG.SYS (see appendix C).

TREE

Displays all the directories and paths on the specified drive. It also has an option to list the files in each directory.

Classification

External

[*d*:][*path*] **TREE** [*drive:*] [/F]

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where TREE is to be found.
path	Specifies the directory where TREE is to be found.
drive	The drive whose directory structure is to be examined.
/F	The files contained in each directory are also to be listed.

Characteristics

For every directory on the specified drive the TREE command will display:

- The directory path starting from the root directory of the drive.
- The subdirectories within the directory.
- The files in the directory (only if **/F** is specified).

Example

IF you enter...	**THEN...**
TREE A: /F	the path to each directory on drive A is displayed along with the subdirectories and files defined within each directory.

TYPE

Displays the contents of the specified file on the video screen.

Classification

Internal

TYPE *pathname*

Where

SYNTAX ELEMENT	MEANING
pathname	The path to the file to be displayed, including the filename. Filename can not be wildcarded.

Characteristics

Use this command to examine a file without modifying it. Press **CTRL NUMLOCK** (or **CTRL S**) to suspend output, press any key to recommence typing. Press **CTRL BREAK**(or **CTRL C**) to terminate output. Press **CTRL PRTSC** (or **CTRL P**) to turn the printer on, press **CTRL PRTSC** again to turn the printer off.

The complete contents of the file, including any non-alphabetic and nonnumeric characters, appear on the screen. As such, the file may appear unreadable.

Tab characters are expanded on the screen to tab stops every eighth column.

Remarks

It is advised that you only type (and print) text files.

VER

Displays the MS-DOS version number.

Classification

Internal

VER

Characteristics

This command displays on your screen the version number of the MS-DOS system you are using.

VERIFY

Sets an internal switch which causes disk writes to be verified.

Classification

Internal

VERIFY [ON|OFF]

Characteristics

This command has the same purpose as the /V switch in the COPY command. If you want to verify that all files are written correctly to disk, you can use the VERIFY command to tell MS-DOS to verify that your files are intact (no bad sectors, for example). MS-DOS will perform a VERIFY each time you write data to a disk. You will receive an error message only if MS-DOS was unable to successfully write your data to disk.

VERIFY ON remains in effect until you change it in a program (by a SET VERIFY system call), or until you issue a VERIFY OFF command to MS-DOS.

If you want to know what the current setting of VERIFY is, enter VERIFY with no options.

VOL

Displays the volume label of the disk in the specified or default drive.

Classification

Internal

VOL [*drive:*]

Where

SYNTAX ELEMENT	MEANING
drive	The drive that contains the disk to be examined.

Characteristics

If the disk does not have a volume label, VOL displays the following message:

> **Volume in drive *x* has no label**

WRITECHK

Set write protection on or off for 48 tpi diskettes in a High Capacity drive.

Classification

External

```
[d:][path] WRITECHK = [YES|NO]
```

SYNTAX ELEMENT	MEANING
d	Specifies the drive where WRITECHK is to be found.
path	Specifies the directory where WRITECHK is to be found.

Characteristics

WRITECHK enables you to protect against accidental High Capacity drive writes on 48 tpi diskettes by entering WRITECHK =YES.

MS-DOS defaults to allowing you to write on 48 tpi diskettes in a High Capacity drive. However, from then on you can only use these 48 tpi diskettes in High Capacity drives; you cannot reliably use them again in Normal Capacity Drives.

If you wish to know the current setting of this protection, enter WRITECHK.

To disable write protection, enter WRITECHK = NO. You will then be able to write on 48 tpi diskettes in High Capacity drives. This is the default.

7. VIDEO FILE EDITOR (EDIT)

ABOUT THIS CHAPTER

This chapter tells you how to use the Video File Editor (EDIT)

CONTENTS

INTRODUCTION	7-1
HOW TO INVOKE THE VIDEO FILE EDITOR	7-1
THE DISPLAY	7-3
THE KEYBOARD	7-4
EDIT	7-7
GENERAL EDITING FUNCTION KEYS	7-8
WINDOW MOVING FUNCTION KEYS	7-16
EXITING AND SAVING FUNCTION KEYS	7-18
COMMANDS AND SEARCHING	7-19
STRING SEARCHES	7-19
COMMANDS	7-21
GOTO	7-21
DELETE	7-22
FILE	7-22

INTRODUCTION

The Video File Editor enables you to create and edit files of text. A text file is a file of records containing printable ASCII characters, and each record is separated from the next by a carriage-return/line-feed pair.

The Video File Editor displays a 21-line ''window'' within which you can perform editing functions via the keyboard. A subset of these functions enables you to move the window to access any part of the file.

In addition to the functions mentioned above the Video File Editor can also perform an extensive set of line editing and cursor moving functions and can operate in overstrike, insert text or command mode. The latter enables a subset of high level commands.

Each text line in a newly created file can contain up to 80 characters. Existing files created by means other than the Video File Editor can be edited with it, even if its lines extend beyond 80 characters. However, all characters after column 80 are overstruck on column 80.

HOW TO INVOKE THE VIDEO FILE EDITOR

EDIT

The EDIT command is used to enter the Video File Editor.

[*d*:][*path*] **EDIT [/B][/T][/R]** *pathname*

Where

SYNTAX ELEMENT	MEANING
d	specifies the drive where EDIT is to be found.
path	specifies the directory where EDIT is to be found.
/B	A backup of the file is to be made when the Video File Editor is entered. This backup is named filename.bak where filename is the same as that specified in the command line.
/T	The size of the file will be minimized by automatically replacing multiple spaces with TAB characters wherever possible.
/R	The read-only option and is used when you only wish to examine the contents of the file. This protects the file from accidental damage while examining it.
pathname	The path to the file to be edited.

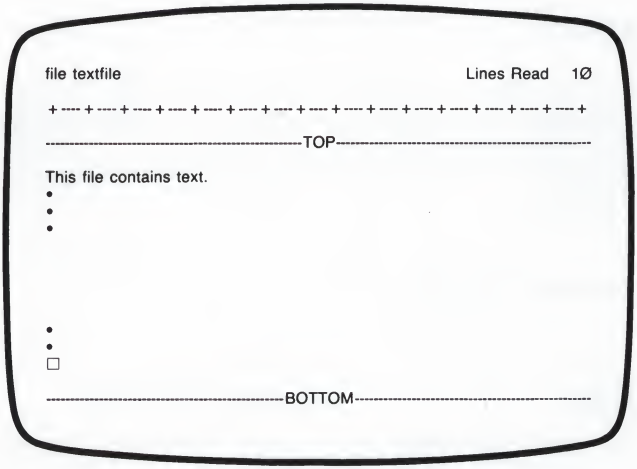
Characteristics

If the file does not already exist the prompt "**OK to Create?**" appears on the screen, to which you must type "**Y**" to create the file.

The Video File Editor is initially in "overstrike" mode. That is, you can enter text and overwrite whatever is already written on the file. The methods of entry into other modes of operation are described later.

THE DISPLAY

Once the Video File Editor has been invoked the monitor shows a display such as:



```
file textfile                                              Lines Read   1Ø
+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+
------------------------------------TOP------------------------------------
This file contains text.
•
•
•



•
•
□
----------------------------------BOTTOM-----------------------------------
```

Fig. 7-1 Video File Editor Screen Layout

Line 1 indicates the file name and the current message.

Line 2 is used for high level commands and search strings and is therefore only used when in command mode. Refer to the section entitled ''Commands and Searching'' for details.

Line 3 shows the tab stop settings (4 character positions per tab).

Lines 4 to 24 contain the text window.

Line 25 is not used.

On entering the Video File Editor the beginning and end of the file are marked by two display lines containing the words TOP and BOTTOM, respectively. The former, known as the TOP bar, always appears immediately before the first line of text in the file. And the BOTTOM bar always appears immediately after the last line of text. They are not actual lines of text and are there merely as markers. The cursor is initially positioned on the TOP bar.

The cursor changes shape when switching between certain modes of editing. It is represented here as underline.

Note that the screen mode for the Video File Editor is 80x25 lines, even if it is invoked from a terminal set to 40x25 lines.

THE KEYBOARD

The keyboard functions in a different manner once the Video File Editor has been invoked. This provides the means by which the required editing functions are entered. The following tables show for each function key, the function name and the key-stroke combination that executes that function.

Using the Numeric Keypad

KEY-STROKE	FUNCTION KEY NAME
HOME	TOP
END	BOTTOM
PGUP	FULL SCREEN UP
PGDN	FULL SCREEN DOWN
←	CURSOR LEFT
→	CURSOR RIGHT
↑	CURSOR UP

▶

KEY-STROKE	FUNCTION KEY NAME
↓	CURSOR DOWN

Using the Function Keys

Note that these function keys are summarized on a template supplied with your system. Keep this template by your keyboard, for quick reference during working sessions.

KEY-STROKE	FUNCTION KEY NAME
F1	COMMAND MODE
SHIFT F1	ABORT
F2	RESTORE LINES
SHIFT F2	DELETE LINE
F3	JOIN LINES
SHIFT F3	SPLIT LINE
F4	END OF LINE
SHIFT F4	START OF LINE

KEY-STROKE	FUNCTION KEY NAME
F5	SAVE
SHIFT F5	SAVE AND EXIT
F6	NEXT LINE
SHIFT F6	ERASE TO END
F7	GOTO MARK
SHIFT F7	INSERT MARK
F8	SEARCH DOWN
SHIFT F8	SEARCH UP
F9	LINE DOWN
SHIFT F9	LINE UP
F10	HALF SCREEN DOWN
SHIFT F10	HALF SCREEN UP

Using Control Keys

KEY STROKE	KEY FUNCTION NAME
CTRL H	BACKSPACE
CTRL I	TAB
CTRL K	ERASE TO END OF LINE
CTRL L	REFRESH
CTRL R	RECALL LINE
← or **BS**	BACKSPACE
INS	INSERT MODE
DEL	DELETE CHAR
ESC	ESCAPE
|← →|	TAB
SHIFT |← →|	REVERSE TAB
ENTER ↵	INSERT LINE or EXECUTE COMMAND

GENERAL EDITING FUNCTION KEYS

The keys whose functions are described below perform general editing functions such as moving the cursor and inserting and deleting text.

CLASS	FUNCTION KEY	MEANING
to move the cursor	↑ (CURSOR UP)	Moves the cursor one line up the screen but keeps the same position within the line. If the cursor was on the second line of the window then the window is moved one line up the file and the cursor remains on the second line.
	↓ (CURSOR DOWN)	Moves the cursor one line down the screen but keeps the same position within the line. If the cursor was on the penultimate line of the window, it stays there and the window is moved down one line.
	← (CURSOR LEFT)	Moves the cursor one character position to the left within the same line.

▶

CLASS	FUNCTION KEY	MEANING
to move the cursor	→ (CURSOR RIGHT)	Moves the cursor one character position to the right within the same line.
	|← →| or **CTRL I** (TAB)	Moves the cursor one tab position (four characters) to the right.
	SHIFT |← →| (REVERSE TAB)	Moves the cursor one tab position (four characters) to the left.
	SHIFT F4 (START OF LINE)	Moves the cursor to the start of the current line.
	F4 (END OF LINE)	Moves the cursor to the character position immediately following the last non-space character in the current line.
to insert text	**INS** (INSERT MODE)	Is entered from overstrike mode. The cursor changes its shape to show that a new mode has been entered. Any character which is subsequently entered is inserted immediately before the cursor position, and the remainder of the text in the line and the cursor are moved one character position to the right. Any character that was in the last character position in the line is discarded. Striking the INSERT MODE key a second time returns the Video File Editor to overstrike mode and the original cursor is restored.

▶

CLASS	FUNCTION KEY	MEANING
to insert text	**ENTER** ↵ (INSERT LINE)	Inserts a blank line immediately after the current line and places the cursor at the beginning of that line. Subsequent text is pushed one line down the screen. If the cursor was already on the bottom line of the screen then the window is moved one line down the file and the blank line is inserted on the last line of the window.
to delete text	← (or **BS**) or **CTRL H** (BACKSPACE)	Moves the cursor one character position to the left and deletes the character under the cursor. Subsequent characters in the line do not move. The deleted characters are replaced with spaces. This function is usually used for correcting typing errors when entering new text.
	DEL (DELETE CHAR)	Deletes the character under the cursor and shifts the subsequent characters in the line one position to the left.
	SHIFT F6 or **CTRL K** (ERASE TO END)	Deletes the contents of the current line from the current cursor position to the end of the line.

▶

CLASS	FUNCTION KEY	MEANING
to delete text	**SHIFT F2** (DELETE LINE)	Deletes the current line and moves subsequent text one line up the screen. The position of the cursor is not changed, it remains in the same column position. The deleted line of text is placed in a holding area called the restore buffer. This action overwrites the previous contents of the restore buffer except where DELETE LINE functions immediately follow each other, in which case subsequent deleted lines are appended to the buffer. This enables you to move a block of text from the file into the buffer, from where it can be reinserted into the same or another file using the RESTORE LINES function.
to restore text	**CTRL R** (RECALL LINE)	Restores the contents of the current line to its original state. The contents restored are those that existed before the cursor was moved to this line. Once the cursor is moved off a particular line the old contents of that line cannot be recalled using this function.
	F2 (RESTORE LINES)	Inserts the contents of the restore buffer into the file starting at the the line below the current cursor position. The cursor is moved to the start of the inserted line(s). The restore buffer itself is not changed. This function is used in conjunction with the DELETE LINE function to move and/or copy blocks of text.

CLASS	FUNCTION KEY	MEANING
to split and join lines of text	**SHIFT F3** (SPLIT LINE)	Divides the current line into two by moving all text under and to the right of the cursor onto the next line. The cursor does not move. Text on subsequent lines is shifted one line down the screen.
	F3 (JOIN LINES)	Combines two lines into one. The text on the subsequent line is placed immediately after the last non-space character on the current line. The cursor does not move. If the current line cannot accommodate the entire text of the next line then only that amount which fits is moved and the remaining text stays on the same line but is moved to the left hand edge of the screen.
to insert a marker	**SHIFT F7** (INSERT MARKER)	Causes a marker to be inserted in the text immediately following the current line. The marker is a dotted line containing the text "MARK". If the MARK line was previously located somewhere else in the text it is moved from where it was to the new position. Note that this is not an actual line of text and will never be written to the file. Its placement is therefore only significant during the current editing session. It is used in conjunction with the GOTO MARK function as a place marker (for details see the section entitled "Window Moving Function Keys"), and in conjunction with the high-level command DELETE (see the section entitled "Commands and Searching").

▶

CLASS	FUNCTION KEY	MEANING
to enter control characters	**ESC** (ESCAPE)	The Video File Editor allows you to enter only the printable ASCII character set (hexadecimal codes 20 to 7E). To force the generation of ''control'' codes (hexadecimal 00 to 1F and 7F) the ESCAPE character must be used. When you type the ESCAPE key a special character (a reverse video symbol) is placed on the screen. This is treated like any any other character except that the following character becomes a control character. This means that only the lower five bits of code are written to the file thereby generating a code in the range 00 to 1F. An exception is the following: to generate a code of 7F you must enter **ESC ?**; this sets the seventh bit. To insert the Escape ASCII character (ESC, hexadecimal 1B), type ESC [

Examples

The following table shows some examples of how text can be modified using the functions discussed above. If assumes a text file called EXAMPLE1 on the B: drive.

STEP	IF you enter...	The screen displays...
	EDIT B: EXAMPLE1	_ The purpose of this text is to act as an example of how to use the editing functions of the Video File Editor
1	CURSOR DOWN DELETE LINE	as an example of how to use the editing functions of the Video File Editor
2	CURSOR UP **ENTER**	_ as an example of how to use the editing functions of the Video File Editor
3	This is **SPACE**	This is _ as an example of how to use the editing functions of the Video File Editor
4	JOIN LINES	This is as an example of how to use the editing functions of the Video File Editor
5	DELETE CHAR DELETE CHAR DELETE CHAR	This is an example of how to use the editing functions of the Video File Editor

▶

STEP	IF you enter...	The screen displays...
6	NEXT LINE	This is an example of how to use the editing functions of the Video File Editor
7	DELETE LINE	This is an example of how to use the Video File Editor
8	RESTORE LINES NEXT LINE	This is an example of how to use the Video File Editor the editing functions of
9	T	This is an example of how to use the Video File Editor The editing function of
10	END OF LINE	This is an example of how to use the Video File Editor The editing functions of_
11	BACKSPACE BACKSPACE	This is an example of how to use the Video File Editor The editing functions _
12	RECALL LINE	This is an example of how to use the Video File Editor the editing functions of
13	SPLIT LINE	This is an example of how to use the Video File Editor the editing functions _ of

▶

STEP	IF you enter...	The screen displays...
14	CURSOR UP	This is an example of how to use the Video File Editor — the editing functions of
15	INSERT LINE	This an example of how to use the Video File Editor — the editing functions of

Note: To delete a character in the 80th column you should move the cursor to that position in overstrike mode and type **SPACE** .

WINDOW MOVING FUNCTION KEYS

The function keys described in the following table enable you to move the window up and down the file.

FUNCTION KEY	MEANING
HOME (TOP)	Moves the window to the top of the text file. The cursor is placed on the top bar of the file.
END (BOTTOM)	Moves the window to the end of the file. The cursor is placed on the last line of text.

FUNCTION KEY	MEANING
PG UP (FULL SCREEN UP)	Causes the window to be moved up the file by 20 lines. This allows one line of overlap between the old and new displays. The cursor remains on the same screen line.
PG DN (FULL SCREEN DOWN)	Causes the window to be moved 20 lines down the file. This allows one line of overlap between the old and new displays. The cursor remains on the same screen line.
SHIFT F10 (HALF SCREEN UP)	Causes the window to be moved half a screen (10 lines) up the file. The cursor remains on the same screen line.
F10 (HALF SCREEN DOWN)	Causes the window to be moved half a screen (10 lines) down the file. The cursor remains on the same screen line.
SHIFT F9 (LINE UP)	Causes the window to be moved one line up the file. The cursor remains on the same screen line.
F9 (LINE DOWN)	Causes the window to be moved one line down the file. The cursor remains on the same screen line.
F6 (NEXT LINE)	Moves the window one line down the file and places the cursor at the start of the next text line.
F7 (GO TO MARK)	Moves the window up or down the file such that the cursor lies on the MARK line.

EXITING AND SAVING FUNCTION KEYS

The function keys described in the following table enable you to exit from the Video File Editor and/or save the file you have been working on.

FUNCTION KEY	MEANING
SHIFT F5 (EXIT AND SAVE)	Causes the revised text to be written back to the file and the Video File Editor to be terminated. The screen is erased and control is returned to MS-DOS.
F5 (SAVE TEXT)	Causes the revised text to be written to the file. The Video File Editor does not terminate.
SHIFT F1 (ABORT)	Causes the Video File Editor to terminate without writing the revised text to the file. If text has been altered or added since starting the editor you are asked to "Confirm Abort?".To confirm press **Y**. Any other action causes the Video File Editor to ignore the ABORT. Control is returned to MS-DOS.

COMMANDS AND SEARCHING

The second line of the screen (above the scale line) is called the editor command line and is used for entering high level commands and search strings.

To enter text on the editor command line you must first press the COMMAND MODE function key. This moves the cursor to the second line. You can now enter text there. All line editing opertions such as INSERT MODE, BACKSPACE and DELETE CHAR - now apply to the editor command line. The RECALL LINE function when used in command mode restores the editor command line to its previous contents. The **ENTER** key performs EXECUTE COMMAND when used in this mode.

Repeating the COMMAND MODE key returns the cursor to the next window without performing any command operation.

STRING SEARCHES

This feature enables you to search the file for a particular combination or characters. Before searching for a text you must enter command mode by striking the COMMAND MODE function key. Then enter the text to be searched for followed by the appropriate function key, as described in the following table:

FUNCTION KEY	MEANING
F8 (SEARCH DOWN)	Searches for the text string starting from the the current cursor position and moving down the file until the first occurrence of the string is encountered. If found, the window and cursor are moved to it.
SHIFT F8 (SEARCH UP)	Searches for the text string starting from the cursor position and moving up the file. If the string is found then the window and cursor are moved to it.

Examples

The following table shows some examples of the use of the searching functions. It assumes a text file called EXAMPLE2 on the B: drive. When you enter at the A> prompt: EDIT B:EXAMPLE2

If you enter on the editor command line...	**Then strike function key...**	**The screen displays...**
	F1	This is an example of how to use the search function keys of the Video File Editor to find a particular combination of characters
func	SEARCH DOWN	This is an example of how to use the search function keys of the Video File Editor to find a particular combination of characters
e **SPACE**of	SEARCH UP	This is an example of how to use the search function keys of the Video File Editor to find a particular combination of characters

COMMANDS

The Video File Editor commands are a set of special commands that enable you to perform a number of high level functions. Before entering a command you must press the COMMAND MODE function key (F1) to move the cursor to the command line. You can then enter the command which is subsequently displayed on the editor command line. To execute the command you must then press the **ENTER** key. If you decide not to execute the command press F1 again to return the cursor to the edit text.

GOTO

This command enables you to move the window to a specific line number in the file.

GOTO *line*

Where

SYNTAX ELEMENT	MEANING
line	A decimal integer that is the desired line number in the file. If this number is greater than the number of lines in the file then the window is moved to the end of the file.

Characteristics

Each line of the text file is automatically numbered. That is, the first line of the file is line 1, the TOP bar is line 0 and the MARK bar does not count.

DELETE

This command removes all text between the current line and the MARK line and places the removed text in the restore buffer from where it can be re-inserted at will. If the MARK line does not exist an error message is given.

DELETE

FILE

The FILE command allows you to suspend processing of the current file and invoke the editor on another file. When editing of the new file is terminated by a SAVE AND EXIT or ABORT function, the old file is recalled at the point at which it was exited.

FILE *pathname*

Where

SYNTAX ELEMENT	MEANING
pathname	The path of the new file to be edited.

Characteristics

The command line option flags (**/B**, **/T** or **/R**) used by the old file remain the same for the new file.

Editing of each file is kept entirely independent except for the restore buffer, which enables the transfer of lines of text from one file to another.

Further files can be entered and edited from the new file using the FILE command. There is no limit to the number of levels that can be created in this way except that the text of all the files invoked must fit into the available memory.

8. LINE EDITOR (EDLIN)

ABOUT THIS CHAPTER

This chapter tells you how to use the Line Editor EDLIN.

CONTENTS

INTRODUCTION **8-1**

HOW TO INVOKE THE EDLIN PROGRAM **8-2**

EDLIN **8-2**

INTER-LINE COMMANDS **8-5**

line (EDIT LINE) **8-6**

A (APPEND LINES) **8-8**

C (COPY LINES) **8-9**

D (DELETE LINES) **8-12**

E (END EDITING) **8-15**

I (INSERT LINES) **8-16**

L (LIST TEXT) **8-20**

M (MOVE LINES) **8-23**

P (PAGE) **8-25**

Q (QUIT EDITING) **8-26**

R (REPLACE TEXT) **8-27**

S (SEARCH TEXT) **8-30**

T (TRANSFER LINES) **8-34**

W (WRITE LINES) **8-35**

INTRA-LINE COMMANDS **8-36**

COPY1 **8-37**

COPYTO **8-38**

COPYLINE **8-40**

SKIP1 **8-41**

SKIPTO **8-42**

KILL **8-44**

INS **8-46**

NEWTEMP **8-47**

INTRODUCTION

The Line Editor (EDLIN) can be used to edit files that contain lines of text, where each line is a maximum of 255 characters, the last of which must be **ENTER**. The files can contain ASCII text or source programs.

Within EDLIN, two types of commands may be used:

- Commands that enable you to perform editing operations on specified lines, a range of lines or an entire file in order to:

  - list, edit, delete and insert lines of text
  - search for a specified text string
  - search for and replace a specified text string
  - create, edit and save new files
  - edit an existing file, save the modified file and keep a back-up of the original file

  These commands are termed ''inter-line'' commands.

- Commands that enable you to perform editing operations within a line of text. These are termed ''intra-line'' commands and utilize the source line facility as described in Chapter 3.

  The control keys described in Chapter 3 can also be used within EDLIN.

In the disk files, the lines of text are not numbered. But when a file is displayed, lines are numbered dynamically. When you create or edit a file, line numbers begin at 1 and are incremented by one through to the end of the file. If you insert new lines between existing lines, all line numbers following the inserted text are automatically incremented by the number of lines inserted. When lines are deleted, all line numbers following the deleted text are decremented automatically by the number of lines deleted. Consequently, lines are always numbered consecutively, starting from 1, through to the last line in the file.

HOW TO INVOKE THE EDLIN PROGRAM

EDLIN

The line editor (EDLIN) is invoked as follows:

[*d*:][*path*] **EDLIN** *filespec*

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where EDLIN is to be found.
path	Specifies the directory where EDLIN is to be found.
filespec	The file specifier of the file to be edited. This must include the drive specifier unless you want to default to the disk containing the EDLIN command, in which case you need to remove any write-protection.

Characteristics

When you invoke EDLIN, the back-up file to the file you specify (if it exists) is erased to ensure there is sufficient room on disk for the output file.

If the file specified exists and is smaller than 75% of the available memory, then the entire file is loaded into memory and the following is displayed:

```
End of input file
*_
```

You can then edit the file. Note that the EDLIN prompt is an asterisk (*), and the cursor is an underline (_).

In the file specified exists and is larger than 75% of the available memory, then only the first part of the file is loaded, until 75% of the available memory is full. The EDLIN prompt (*) and cursor (_) will then appear but not the ''End of input file'' message. You can then edit that part of the file loaded into memory. To access unloaded lines you must use the Write Lines and Append Lines commands described later in this chapter.

If the specified file does not exist on the drive then a new file is created with the specified name. But note that the drive you wish the output file to be written to must be specified when you invoke EDLIN, otherwise the output file will be written to the default drive. The following message is displayed:

```
New file
*_
```

You can then begin to create the file.

There are two edit commands that can be used to terminate the edit session:

- End Edit, which terminates EDLIN, renames the input file filename.BAK and writes the edited file in memory to the output file which is given the same name as the input file. See the ''E (END EDIT)'' command later in this chapter.

- Quit Edit, which terminates EDLIN without creating a back-up or an output file. The input file remains unchanged. See the ''Q (QUIT EDIT)'' command later in this chapter.

Note that a file with the extension .BAK cannot be edited. Any attempt to do so will generate the message:

Cannot edit .BAK file--rename file

You must rename the file using the RENAME command (See Chapter 6), then invoke EDLIN on the renamed file.

If, when attempting to create a new file, the following message appears:

No room in directory for file

then either:

- the file directory is full, or

- you have specified an illegal disk drive or file name.

The latter can be checked by examining the command line. (If the command line is no longer on the screen it can be recalled using the **F3** (COPYLINE) edit key.) To check the former you can run the CHKDSK command on the specified disk drive. See Chapter 6 for details.

INTER-LINE COMMANDS

This section describes the EDLIN commands that operate on entire lines of text.

The lines you wish to work on may be specified either by entering a line number as a parameter to the command, or by entering a period (.). The latter indicates that the ''current line'' is to be worked on.

The current line is the location of the last change to the file. It is not necessarily the last line displayed. The current line is indicated by an asterisk between the line number and the first text character. For example:

```
15:* This is the current line
```

Each command description summarizes the purpose of the command, defines the command syntax and explains each syntax element. This is followed, for each command, by a detailed account of the command characteristics and some working examples.

Remarks

1. Commands can be entered in either upper or lower case
2. Command keywords and command parameters can be separated from each other by spaces or commas for readability but need not be, except where two line numbers are entered as parameters, in which case they must be separated by a comma or space. For brevity the syntax of this chapter will always indicate comma where separation is obligatory, but note that a space can alternatively be used
3. Commands only become effective after entering **ENTER**
4. If you make a syntax error when entering a command the message ''Entry Error'' will be displayed. You must re-enter the command using the correct syntax

line (EDIT LINE)

Enables you to edit a specified line.

[*line* | .]

Where

SYNTAX ELEMENT	MEANING
line	The number of the line to be edited.
.	The current line is to be edited.

Characteristics

When you enter a line number followed by **ENTER** EDLIN displays the line number and the corresponding text, then, on the next screen line, reprints the line number followed by the EDLIN prompt (*) and the cursor (__) . The displayed line serves as the source line and is ready to be edited.

If, instead of a number, you enter a period (.), the current line is displayed and ready for editing. If you enter **ENTER** without a line number or a period, then the line immediately following the current line is displayed, unless the current line is the last one in the file, in which case the edit prompt (*) will re-appear.

To edit the line you can use any of the control or edit (intraline) keys, or re-enter the entire line by entering text.

If you wish to abort the operation without changing the source line, press **CTRL C** . Pressing **ENTER** with the cursor at the start of the line also aborts the operation without changing anything.

To save the edited line and overwrite the original line, type with the cursor at the end of the edited line. If you type with the cursor in any position other than the start or end of the line, text to the left of the cursor will be written to the file in memory, but all text under and to the right of the cursor will be lost.

Example

Assuming that the contents of the current edit file are as follows:

```
1: This file demonstrates how
2: the line command can
3: be used to edit line
4: four.
```

IF you type...	**THEN EDLIN displays...**
4 **ENTER**	4:*four. 4:*_
INS number **SPACE INS** **F3** **ENTER**	4:*four. 4:*number four._ *

A (APPEND LINES)

Adds lines from the input file on disk to that part of the file currently in memory.

[*n*] **A**

Where

SYNTAX ELEMENT	MEANING
n	The number of lines to be added to the file in memory from the input file on disk.

Characteristics

This command is only useful for files that are too large to fit into the available memory.

When EDLIN is invoked on a file that is too large to fit into memory, it loads as many lines as possible (as much as will fit into 75% of the available memory). Before using the Append command it is therefore necessary to write some lines of text to the output file on disk by means of the Write Lines command (see later).

If you enter A without a parameter, lines are appended from the disk file until the available memory is 75% full, or until there are no more lines to append.

Any attempt to read beyond the end of the disk file will yield the following message:

End of input file

Example

IF you enter...	THEN...
100 A	the next 100 lines of the input file on disk are read into memory and appended to that part of the file that is already there.

C (COPY LINES)

Copies a range of lines to a specified line.

[*line-a*] , [*line-b*] , *line-c*[, *count*]**C**

Where

SYNTAX ELEMENT	MEANING
line-a	The first line in the range to be copied.

SYNTAX ELEMENT	MEANING
line-b	The last line in the range to be copied.
line-c	The line at which the copied lines are to start.
count	The number of times the range is to be copied.

Characteristics

If *line-a* is omitted, then the first line defaults to the current line.

If *line-b* is omitted, then the last line defaults to the current line.

If both *line-a* and *line-b* are omitted, then only the current line is copied.

Following the copy operation, lines that previously followed *line-c* are moved to follow the copied block.

If *line-c* is beyond the current end of file then the lines are copied to line numbers contiguous to the end of file.

line-b must be greater than or equal to *line-a*.

On completion *line-c* becomes the current line.

Examples

Assuming the contents of the current edit file are as follows:

```
1:*This is a sample file
2: to demonstrate the use
3: of the Copy lines command.
```

IF you enter...	THEN the edited file becomes...
1,3,4C	1: This is a sample file 2: to demonstrate the use 3: of the Copy lines command. 4:*This is a sample file 5: to demonstrate the use 6: of the Copy lines command
,,27,3C	1: This is a sample file 2: to demonstrate the use 3: of the Copy lines command. 4: This is a sample file 5: to demonstrate the use 6: of the Copy lines command. 7:*This is a sample file 8: This is a sample file 9: This is a sample file
2,3,8C	1: This is a sample file 2: to demonstrate the use 3: of the Copy lines command. 4: This is a sample file 5: to demonstrate the use 6: of the Copy lines command. 7: This is a sample file 8:*to demonstrate the use 9: of the Copy lines command. 10: This is a sample file 11: This is a sample file

D (DELETE LINES)

Deletes all lines within a specified range.

[*line-a*][,*line-b*] **D**

Where

SYNTAX ELEMENT	MEANING
line-a	The first line in the range to be deleted.
line-b	The last line in the range to be deleted.

Characteristics

If *line-a* is omitted, then the first line defaults to the current line.

If *line-b* is omitted, then the specified line only is deleted.

If D is entered alone, then only the current line is deleted.

After the command has been executed the numbers of the lines following the deleted section are changed to follow the numbers of the lines preceding the deleted section.

The line immediately following the deleted section becomes the current line.

line-b must be greater than (or equal to) *line-a*

Example

Assuming the contents of the current edit file are as follows:

```
 1: This is a sample file
 2: to demonstrate the use
 3: of the Delete command.
 4: "line-a D" deletes just the specified line
 5: if ",line-b D" is specified then
 6: all lines from the current line
 7: to line-b, inclusive, are deleted.
 8: Specifying both line-a and
 9: line-b causes that range
10: to be deleted.
11: Specifying D alone deletes the current line.
12: The line subsequent to the
13: deleted line(s) becomes the
14: current line.
```

IF you enter...	THEN the edited file becomes...
8,10 D	1: This is a sample file 2: to demonstrate the use 3: of the Delete command. 4: "line-a D" deletes just the specified line 5: if ",line-b D" is specified then 6: all lines from the current line 7: to line-b, inclusive, are deleted. 8:*Specifying D alone deletes the current line. 9: The line subsequent to the 10: deleted line(s) becomes the 11: current line.

▶

IF you enter...	THEN the edited file becomes...
4 D	1: This is a sample file 2: to demonstrate the use 3: of the Delete command. 4:*if ",line-b D" is specified then 5: all lines from the current line 6: to line-b, inclusive, are deleted. 7: Specifying D alone deletes the current line. 8: The line subsequent to the 9: deleted line(s) becomes the 10: current line.
,6 D	1: This is a sample file 2: to demonstrates the use 3: of the Delete command. 4:*Specifying D alone deletes the current line. 5: The line subsequent to the 6: deleted line(s) becomes the 7: current line.
D	1: This is a sample file 2: to demonstrate the use 3: of the Delete command. 4:*The line subsequent to the 5: deleted line(s) becomes the 6: current line.

E (END EDITING)

Exits EDLIN and saves the edited file on disk.

E

Characteristics

The edited file is written to the drive selected when EDLIN was invoked, or to the default drive if no drive was specified. The input file is renamed ''filename.BAK''. If the file was created during the editing session, no back-up file is created.

You must make sure that enough free space is available on disk to take the output file, otherwise only a portion (at most) will be saved. The remainder will be lost and the message:

Disk full--write not completed

will be displayed, and EDLIN will exit.

Example

IF you enter...	THEN...
E	The current edit file is saved on the diskette specified when EDLIN was invoked. The input file is renamed ''filename.BAK'', and EDLIN is exited.

I (INSERT LINES)

Allows you to insert lines of text before the specified line number.

[*line*| . | #] I

Where

SYNTAX ELEMENT	MEANING
line	The number of the line before which subsequently entered text is to be inserted.
.	Subsequent text is to be inserted before the current line.
#	Subsequent text is to be appended to the file.

Characteristics

If a line number is entered, then subsequently entered text is inserted immediately before the specified line. If, instead of a line number you enter a period (.), or you enter I on its own, then subsequent text is inserted before the current line. Moreover, if you enter # instead of the line number, text will be appended to the file.

EDLIN remains in insert mode until either **CTRL Z** or **CTRL C** is entered. While you are in insert mode, successive line numbers appear each time **ENTER** is pressed.

When you exit insert mode, the line immediately following the last inserted line becomes the current line. The line numbers of this and all subsequent lines are automatically incremented by the number of lines inserted.

When a file is created, you must enter I before entering text. In this case the line number is 1.

Example

Assuming that the contents of the current edit file are as follows:

```
1: This is a sample file
2: to demonstrate
3: using the Insert Lines
4: command
```

IF you enter..	THEN...
3 I	EDLIN enters insert mode and displays: **3:*_**
	EDLIN displays:
how lines of text	**3:*how lines of text**
can be inserted into a file	**4:*can be inserted into a file**
at a specified line number	**5:*at a specified line number**
	6:*_

IF you enter...	THEN...
CTRL Z	EDLIN exits insert mode.
L	The List command (see later) is invoked and EDLIN displays: **1: This is a sample file** **2: to demonstrate** **3: how lines of text** **4: can be inserted into a file** **5: at a specified line number** **6:*using the Insert Lines** **7: command**
I or .I	EDLIN enters insert mode and displays: **6:*_**
or before the current line	EDLIN displays: **6: or before the current line** **7:*_**
CTRL Z	EDLIN exits insert mode.
L	The List command is invoked and EDLIN displays: **1: This is a sample file** **2: to demonstrate** **3: how lines of text** **4: can be inserted into a file** **5: at a specified line number** **6: or before the current line** **7:*using the Insert Lines** **8: command**

▶

IF you enter...	THEN...
# I or 9	EDLIN enters insert mode and displays: 9:*_
and how lines of text can be appended to a file.	EDLIN displays: **9: and how lines of** **10: text can be appended** **11: to a file.** 12:*
CTRL Z	EDLIN exits insert mode.
L	The List command is invoked and EDLIN displays: **1: This is a sample file** **2: to demonstrate** **3: how lines of text** **4: can be inserted into a file** **5: at a specified line number** **6: or before the current line** **7: using the Insert Lines** **8: command** **9: and how lines of** **10: text can be appended** **11: to a file.**

L (LIST TEXT)

Displays a specified range of lines.

[*line-a*][, *line-b*] **L**

Where

SYNTAX ELEMENT	MEANING
line-a	The first line in the range to be listed.
line-b	The last line in the range to be listed.

Characteristics

If you specify both *line-a* and *line-b*, then the entire range of lines is displayed, unless this is in excess of 23 lines, in which case the display starts from *line-a*, but this and subsequent lines are scrolled off the top of the screen until *line-b* appears on the 23rd line.

If *line-a* is omitted, but *line-b* is specified, then the display starts 11 lines before the current line and ends at *line-b*. If this is more than 23 lines, the screen scrolls down the file until *line-b* appears on the 23rd line.

If *line-b* is omitted, but *line-a* is specified, then 23 lines are displayed, starting from *line-a*.

If you enter L on its own, then the current line appears on the center line of the screen (unless the current line is less than line 12), with the preceding 11 lines displayed before it, and the subsequent 11 lines displayed after it. If the current line is before line 12, EDLIN displays the first 23 lines.

Example

Assuming the contents of the current edit file are as follows:

```
 1: This is a sample file
 2: to demonstrate the
 3: use of the List
 4: command
    .
    .
    .
14:*This is the current line
    .
    .
    .
23: The List command can be
24: used to examine
25: different parts of the
26: file, up to 23 lines
27: at once.
```

IF you enter...	THEN EDLIN displays...
2,4 L	**2: to demonstrate the** **3: use of the List** **4: command**

▶

IF you enter...	THEN EDLIN displays...
,24 L	3: use of the List 4: command . . . 14:*This is the current line . . . 23: The List command can be 24: used to examine
24 L	24: used to examine 25: different parts of the 26: file, up to 23 lines 27: at once.
L	3: use of the List 4: command . . . 14:*This is the current line . . . . 24: used to examine 25: different parts of the

M (MOVE LINES)

Moves a range of lines to a specified line.

[*line-a*] , [*line-b*] , [*line-c*] **M**

Where

SYNTAX ELEMENT	MEANING
line-a	The first of the range of lines to be moved.
line-b	The last of the range of lines to be moved.
line-c	The line to which the text is to be moved.

Characteristics

If *line-a* is omitted, then the first line defaults to the current line.

If *line-b* is omitted, then the last line defaults to the current line.

line-b must be greater than or equal to *line-a*.

If *line-a* is omitted, *line-b* can be specified as the relative number of lines forward of the current line, by preceding the number with a + .

Following the move, lines are renumbered depending on the direction of the move. For instance, moving lines 10 to 20 to line 100 would effectively delete lines 10 to 20 (thereby causing all subsequent lines to be moved up the file 11 lines), then the moved lines would become lines 79 to 99.

The first of the moved lines becomes the current line.

Examples

Assuming the contents of the current edit file are as follows:

```
  1: This is a sample file
  2: to demonstrate the use
  3: of the Move lines command.
  4: New first line
   .
   .
100: Next line.
```

IF you enter...	THEN the edit file becomes...
1,3,100 M	1: New first line . . . 97: This is a sample file 98: to demonstrate the use 99: of the Move lines command. 100: Next line.

P (PAGE)

Lists a specified range of lines.

[*line-a*][, *line-b*] **P**

Where

SYNTAX ELEMENT	MEANING
line-a	The first of the range of lines to be displayed.
line-b	The last of the range of lines to be displayed.

Characteristics

If *line-a* is omitted, then the first line defaults to the line following the current line.

If *line-b* is omitted 23 lines are displayed.

The last line displayed becomes the current line.

The difference between the Page command and the List command is that the Page command changes the current line.

Q (QUIT EDITING)

Quits the editing session but does not save any changes you have made.

```
Q
```

Characteristics

After entering the Quit command EDLIN replies with the message:

Abort edit (Y/N)?_

Pressing **N** or any key other than **Y** or **CTRL C** continues the editing session. Pressing **Y** terminates the editing session. No BAK file is created, and any changes made during the editing session are lost. The file on disk remains exactly as it was when EDLIN was invoked. Note also that any previous BAK file is also lost since the current BAK file is always deleted when EDLIN is invoked.

Example

IF you enter...	THEN...
Q	the editing session is terminated without saving the changes made during the editing session.

R (REPLACE TEXT)

Searches a specified range to replace all occurrences of one string with another string.

[*line-a*][,*line-b*] [**?**] **R** *string-a*[**CTRL Z** *string-b*]

Where

SYNTAX ELEMENT	MEANING
line-a	The number of the first line in the range on which the Replace Text command is to be executed.
line-b	The number of the last line in the range.
?	For each occurrence of the specified string the **O.K.?** prompt appears, enabling you to accept or reject the replacement.
string-a	The string of characters that are to be replaced.
string-b	The string of characters that are to replace *string-a*.

Characteristics

For each line in which a replacement occurs the modified line is displayed on the screen. If you entered the ? parameter then the prompt:

O.K.?

will appear after each replacement. You must then enter **Y** or **ENTER** to confirm the replacement, or strike any other key to reject it. In either case the search will recommence for the next occurrence of *string-a*. If ? is not specified, all occurrences of *string-a* will be replaced by *string-b* without confirmation.

If you omit *string-b*, then all occurrences of *string-a* are deleted.

If you omit *line-a* then the search will begin from line 1. If you omit *line-b* then the search will continue to the end of the file in memory. If neither *line-a* nor *line-b* is entered, then the entire file in memory will be searched and modified.

Once all replacements have been made, the Replace Text command terminates and the last line in which *string-a* occurred becomes the current line.

If the replacement string causes a line to expand beyond the limit of 254 characters then the following message is displayed:

Line too long

Example

Assuming that the contents of the current edit file are as follows:

```
1: This is a sample file
2: to demonstrate the
3: use of the Replace Text
4: command.
5: Using this command a
6: specified group of characters
7: can be replaced by
8: another group of characters
9:*and can be deleted entirely.
```

IF you enter...	THEN... .
5,8 R group of characters **CTRL Z** string	all occurrences of ''group of characters'' in the range line 5 to line 8 are replaced with ''string''. EDLIN displays: **6: specified string** **8: another string** and line 8 becomes the current line
? Rand **CTRL Z** or	each occurrence of ''and'' within the entire file can optionally be replaced with ''or''. EDLIN displays: **4: commor.** **O.K.?**
N	**5: using this commor a** **O.K.?**
N	**9: or can be deleted entirely?** **O.K.?**
Y	*_ and line 9 becomes the current line
L	the List command is executed to display the file and enable you to see the changes you have made. EDLIN displays: **1: This is a sample file** **2: to demonstrate the** **3: use of the Replace Text** **4: command.** **5: Using this command a** **6: specified string** **7: can be replaced by** **8: another string** **9:*or can be deleted entirely.**

S (SEARCH TEXT)

Searches a specified range of lines for a specified string.

[*line-a*][, *line-b*] [**?**] **S** *string*

Where

SYNTAX ELEMENT	MEANING
line-a	The number of the line from which the search is to start.
line-b	The number of the last possible line to be searched.
?	On finding a matching string the: **O.K.?** prompt is to be displayed, thereby enabling you to accept or reject the particular occurrence.
string	The string of characters to be searched for.

Characteristics

The command searches the range *line-a* to *line-b* for the specified string. If *line-a* is not specified then the search begins from the line after the current line. If *line-b* is not specified then the last line in the range is the last line of the file in memory.

When a matching string is found, the corresponding line is displayed. Then if the **?** parameter is not specified, the search is terminated and the displayed line becomes the current line. If **?** is specified, then the displayed line will be followed by the prompt:

O.K.?

To accept the string you must enter either **Y** or **ENTER** . The search will then terminate and the displayed line becomes the current line. Striking any other key, however, will re-commence the search for the next occurrence of the string.

If no matching string is found, or if **?** is specified and all matching strings are rejected, the message:

Not found

is displayed.

Example

Assuming that the contents of the current edit file are as follows:

```
 1: This is a sample file to demonstrate
 2: The use of the Search Text command.
 3: The search can either display the
 4: first occurrence of a specified string
 5: and terminate, or, if specified
 6: to do so, it will enable you to
 7: interactively examine each
 8: occurrence of a string allowing
 9: you to confirm or reject the string.
10: Once a string is accepted the
11:*search terminates.
```

IF you enter...	THEN...
2,5 Sstring	the Search Text command will search lines 2 to 5 of the file, inclusive, for the first occurrence of the string ''string'', then display the following: **4: first occurrence of a specified string** and terminate. Line 4 becomes the current line.
,8 Sstring	the search will be made on lines 5 (one after the current line) to 8, inclusive, for the first occurrence of ''string''. EDLIN will display: **8: occurrence of a string allowing** and the search will terminate with line 8 as the current line.
1? Sstring	the Search Text command searches the file for the first occurrence of ''string''. The search starts from line 1. The result is: **4: first occurrence of a specified string** **O.K.?**
N	The string is rejected and the search continues for the next occurrence. The result is: **8: occurrences of a string allowing** **O.K.?**
Y	The search is terminated and line 8 becomes the current line.

▶

IF you enter...	THEN...
Ssample file	the Search Text command searches for the string ''sample file'' starting from line 9 (one after the current line), up to the end of the file. The string is not found, hence the message: **Not found** is displayed. The search terminates and line 8 remains the current line.

T (TRANSFER LINES)

Inserts an entire file before a specified line of the current edit file.

[*line*] **T** [*filespec*]

Where

SYNTAX ELEMENT	MEANING
line	The number of the line before which the file is to be inserted.
filespec	The file to be inserted.

Characteristics

If the line parameter is omitted then the current line is assumed.

The specified file must be in the same directory as the edit file.

W (WRITE LINES)

Writes a specified number of lines from the file being edited in memory to the output file on disk.

[*n*] **W**

Where

SYNTAX ELEMENT	MEANING
n	The number of lines to be written to diskette, starting from line 1. If this parameter is omitted, then lines of text are written to the output file until the available memory is 25% full.

Characteristics

The Write Lines command is used in conjunction with the Append Lines command when editing files that are too large to fit into the available memory. Lines written to the output file are deleted from memory, and the remaining lines renumbered, starting from line 1. This leaves space available at the end of the file in memory for additional lines to be read from the input file on disk using the Append Lines command.

INTRA-LINE COMMANDS

The intra-line commands are executed using the special editing keys that can be used to perform edits within the current line taking advantage of the source line facility. They enable you to:

- copy one character from the source line to the current line (COPY1)
- copy a specified portion of the source line to the current line (COPYTO)
- copy all remaining characters in the source line to the current line (COPYLINE)
- delete a specified character in the source line (SKIP1)
- delete a specified portion of the source line (SK!PTO)
- kill the current input and delete the source line (KILL)
- enter insert mode to insert text into the current line (INS)
- exit insert mode (enter overstrike mode) (INS)
- make the current line the source line (NEWTEMP)

Moreover, you can also use the MS-DOS control keys when in EDLIN.

For details about control keys refer to Chapter 3.

The remainder of this section describes each of the intra-line commands in turn.

Before starting to edit the current line, a copy of the current line exists in the source line. You then begin editing the current line by entering an edit line by entering text and by using the intra-line commands. Not until you complete the edit line by pressing **ENTER** does the edit line replace the contents of the current line.

You can select a line to work on using the line command described in the previous section.

COPY1

Copies one character from the source line to the edit line.

F1

Characteristics

Pressing the **F1** key copies one character from the source line to the edit line. Insert mode, if active, is automatically turned off.

Example

Assuming that the line to be edited is displayed as follows:

```
1:*This is the COPY1 command
1:*_
```

IF you enter...	THEN...
F1	the first character is copied from the source line into the edit line thus: **1:*This is the COPY1 command** **1:*T_**
F1	the next character is copied from the source line into the edit line thus: **1:*This is the COPY1 command** **1:*Th_**

COPYTO

Copies up to a given character from the source line to the edit line.

F2 *character*

Where

SYNTAX ELEMENT	MEANING
character	A character whose first occurrence in the source line will terminate the copy operation. If the character does not appear in the source line nothing will be copied.

Characteristics

Pressing the **F2** key copies all characters up to but not including a given character from the source line to the edit line. The cursor is moved to the position of the given character. The given character is not displayed.

Example

Assuming that the line to be edited is displayed as follows:

```
1:*This is the COPYTO command
1:*_
```

IF you enter...	THEN...
F2 c	the characters "This is the COPYTO " are copied from the source line to the edit line thus: 1:*This is the COPYTO command 1:*This is the COPYTO _

COPYLINE

Copies the source line to the edit line.

F3

Characteristics

Pressing the **F3** key copies all remaining characters from the source line to the edit line regardless of cursor position. Following the copy, the cursor is positioned after the last character on the line. Insert mode, if active, is automatically turned off.

Examples

Assuming that the line to be edited is displayed as follows:

```
1:*This is the COPYLINE command
1:*_
```

IF you enter...	THEN...
F3	all remaining characters are copied from the source to the edit line thus: 1:*This is the COPYLINE command 1:*This is the COPYLINE command_

SKIP1

Skip one character in the source line.

DEL

Characteristics

Pressing the **DEL** key skips over one character in the source line without copying it to the edit line. It does not affect insert mode.

Example

Assuming that the line to be edited is displayed as follows:

```
1:*This is the SKIP1 command
1:*
```

IF you enter...	THEN...
DEL	you skip the letter ''T'' in the source line.
F3	the remaining characters are copied from the source line to the edit line thus: 1:*This is the SKIP1 command 1:*his is the SKIP1 command_

SKIPTO

Skip to specified character in the source line.

F4 *character*

Where

SYNTAX ELEMENT	MEANING
character	The character in the source line that terminates the string that is to be skipped.

Characteristics

Pressing the **F4** key causes characters in the source line to be skipped from the edit character up to but not including the first occurrence of the given character. If the source line does not contain the given character then no characters are skipped. Nothing is copied to the edit line by this command. Insert mode remains unaffected.

Examples

Assuming that the line to be edited is displayed as follows:

```
1:*This is the SKIPTO command
1:*_
```

IF you enter...	THEN...
F4 c	all characters in the source line up to the first "c" are skipped over.
F3	the remaining characters in the source line are copied to the edit line thus: **1:*This is the SKIPTO command** **1:*command_**

KILL

Clear the edit line.

ESC

Characteristics

Pressing the **ESC** key clears the edit line, but the source line remains unchanged. The KILL command also displays a back-slash (\) and inserts a carriage return and a line-feed. The cursor is placed immediately under the first character of the terminated line. You can then begin again to edit the line. Insert mode is turned off by this command.

Examples

Assuming your current and edit lines are displayed as follows:

```
1:*This is the KILL command
1:*This is the_
```

IF you enter...	THEN...
ESC	the edit line is emptied thus: **1:*This is the KILL command** **1:*This is the\** —
F3	the source line is copied to the edit line thus: **1:*This is the KILL command** **1:*This is the\** **This is the KILL command**

INS

Enters/exits insert mode.

INS

Characteristics

Pressing the **INS** key enters or exits insert mode. On entering insert mode, subsequently entered characters will be inserted before the character under the cursor when insert mode was entered.

Examples

Assuming the line to be edited is displayed as follows:

```
1:*This is INS command
1:*This is _
```

IF you enter...	THEN...
INS the	the characters "the" are inserted in the edit line thus: **1:*This is INS command** **1:*This is the _**
INS **F3**	insert mode is switched off and the remainder of the source line is copied to the edit line thus: **1:*This is INS command** **1:*This is the INS command_**

NEWTEMP

Creates a new source line by copying the edit line to the source line.

F5

Characteristics

Pressing the **F5** key copies the edit line to the source line. The original contents of the source line are deleted. An "@" sign appears at the end of the edit line and a carriage return line-feed is inserted. The edit line is also cleared and and insert mode is exited.

Example

Assuming the source and edit lines are displayed as follows:

```
1:*This is the NEWLINE command
1:*_
```

and you want to change the source line to read "This is the NEWTEMP command":

IF you enter...	THEN...
F2 L	all characters up to the first "L" are copied from the source line to the edit line thus: **1:*This is the NEWLINE command** **1:*This is the NEW__**
TEMP	the next four characters in the source line are replaced in the edit line by the characters "TEMP" thus: **1:*This is the NEWLINE command** **1:*This is the NEWTEMP__**
F3	the remaining characters in the source line are copied to the edit line thus: **1:*This is the NEWLINE command** **1:*This is the NEWTEMP command__**
F5 **F3**	the contents of the source line are replaced with those of the edit line. The display appears thus: **1:*This is the NEWLINE command** **1:*This is the NEWTEMP command@** **This is the NEWTEMP command**

9. THE LINKER

ABOUT THIS CHAPTER

This chapter describes the LINK utility.

CONTENTS

INTRODUCTION **9-1**

TEMPORARY FILES **9-1**

CHANGING DISKETTES **9-2**

SEGMENTS, GROUPS AND CLASSES **9-2**

INTERACTIVE ENTRY **9-3**

COMMAND LINE ENTRY **9-10**

AUTOMATIC RESPONSE FILE ENTRY **9-13**

INTRODUCTION

MS-LINK is an executable program available on your MS-DOS system diskette. It links separate object modules that are the output of the assembler or a compatible compiler, resolving external references by searching multiple library files. Its output is a relocatable run file, along with a list file that shows external references and error messages.

To run MS-LINK you must provide appropriate object, run, list and library file parameters. In addition, you may enter switches that modify the way in which MS-LINK processes your input. Parameters and switches are fully described in the section ''Interactive Entry''.

There are three related methods of running MS-LINK. These are described later in this chapter in the following sections:

- Interactive Entry
- Command Line Entry
- Automatic Response File Entry

Interactive entry is the primary method and its section contains all the information common to the three methods.

TEMPORARY FILES

MS-LINK uses available memory for the link session. If the files to be linked create an output file that exceeds available memory, MS-LINK creates a temporary file on the default drive and names it VM.TMP. If MS -LINK needs to create VM.TMP, it displays the message:

VM.TMP has been created.
Do not change diskette in drive *x*:

Once this message is displayed, you must not remove the diskette from the default drive until the link session ends. If the diskette is removed, the operation of MS-LINK is unpredictable, and MS-LINK might return the error message:

Unexpected end of file on VM.TMP

MS-LINK uses VM.TMP as virtual memory. The contents of VM.TMP are subsequently written to the file name following the run file prompt. VM.TMP is a working file only and is deleted at the end of the linking session.

If the default drive already has a file by the name of VM.TMP, it will be deleted by MS-LINK and a new file will be allocated; the contents of the previous file are destroyed. You should therefore avoid using VM.TMP as one of your own file names.

CHANGING DISKETTES

You may wish to change diskettes during the link operation. For example, if MS-LINK cannot find an object file on the specified diskette it prompts you to change diskettes instead of aborting the session. Or if you enter the /PAUSE switch, MS-LINK pauses and prompts you to change diskettes before creating the run file. You may change diskettes when prompted except in the following cases:

- When the diskette you wish to change has a VM.TMP file created on it (see the previous section)

- When you have requested a list file on the diskette you wish to change

SEGMENTS, GROUPS AND CLASSES

Some of the terms used in this chapter are explained below to help you understand how MS-LINK works. Generally, if you are linking object modules compiled from BASIC, Pascal, or any high-level language, you will not need to know these terms. If you are writing and compiling programs in assembly language, however, you will need to understand MS-LINK and the definitions described below.

In MS-DOS, memory can be divided into segments, classes, and groups. For example:

Group Contents	Segment Names	Segment Class Names
Segment 1	PROG.1	CODE
Segment 2	PROG.2	CODE
Segment 12	PROG.3	DATA

Note that segments 1, 2, and 12 have different segment names but may or may not have the same segment class name. Segments 1, 2, and 12 form a group with a group address of the lowest address of segment 1 (that is, the lowest address in memory).

Each segment has a segment name and a class name. MS-LINK loads all segments into memory by class name from the first class encountered to the last. All segments assigned to the same class are loaded into memory contiguously.

During processing, MS-LINK references segments by their addresses in memory. MS-LINK does this by finding groups of segments.

A group is a collection of segments that fit within a 64K byte area of memory. The segments do not need to be contiguous to form a group. The address of any group is the lowest address of the segments in that group. At link time, MS-LINK analyzes the groups, then references the segments by the address in memory of that group. A program may consist of one or more groups.

If you are writing in assembly language, you may assign the group and class names in your program. In high-level languages (BASIC, COBOL, FORTRAN, Pascal), the naming is done automatically by the compiler.

INTERACTIVE ENTRY

With interactive entry MS-LINK prompts you for each parameter in turn.

Type the following:

[*d*:][*path*] **LINK**

Four prompts appear, one at a time, requesting the appropriate parameters. These are summarized in the following table.

PROMPT	RESPONSE
Object Modules [OBJ]:	*objfile*[+ *objfile*]...[*switch*]...
Run File [objfile.EXE]:	[*runfile*][*switch*]...
List File [NUL.MAP]:	[*listfile*][*switch*]...
Libraries [.LIB]:	[*libfile*[+ *libfile*]...][*switch*]...

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where LINK is to be found.
path	Specifies the directory where LINK is to be found.
objfile	The file specification of an object module to be linked. Such specifications must be separated from each other with a plus sign (+) or a **SPACE**. The default file extension is .OBJ. If any extension is different from .OBJ it must be specified; otherwise it may be omitted. Segments are loaded by class name, from the first class encountered to the last. The order in which you list object files is therefore significant.

SYNTAX ELEMENT	MEANING
runfile	The file specification of the run (executable) file that results from the link session. All run files receive the extension .EXE. Any other extension you may enter is ignored. The default run file name is the first object file name entered.
listfile	The file specification of the list file that contains an entry for each segment in the object files and its offset in the run file. The default list file name is the NUL file. The default extension is .MAP.
libfile	The file specification of each library. Up to eight libraries may be searched; separate each specification with a plus sign (+) or a **SPACE** . The default is standard library search. The default extension is .LIB. Libraries are searched in the order they are listed.
switch	The name of any of the switches described in the table below. You may enter any number of switches, at the end of any number of prompt responses (before pressing **ENTER**). Switches may be abbreviated to the / together with the first letter or any sequential substring starting with the first letter.

Note

That [*d*:][*path*] may precede any file name mentioned.

SWITCH	MEANING
/DSALLOCATE (or /D)	All data defined to be in DGROUP is loaded at the high end of the group. If the switch is not used MS-LINK loads all data at the low end of the group. At runtime the data space pointer is set to the lowest possible address, allowing the entire storage to be used. Use of the /DSALLOCATE switch in combination with the default load low (that is, the /HIGH switch is not used) permits the user application to allocate dynamically any available memory below the area specifically allocated within DGROUP, yet to remain addressable by the same data space pointer. This dynamic allocation is needed for Pascal and FORTRAN programs. The maximum amount of memory that can be allocated by the application is 64K (or the amount actually available) minus the allocated portion of DGROUP.
/HIGH (or /H)	MS-LINK loads the run file as high as possible in memory. If /HIGH is not specified the run file is loaded as low as possible. DO NOT USE THIS SWITCH WITH PASCAL OR FORTRAN PROGRAMS.
/LINENUMBERS (or /L)	MS-LINK includes in the list file the line numbers and addresses of the source statements in the input modules. If /LINENUMBERS is not specified, line numbers and addresses are not included. (Not all compilers produce object modules that contain line number information. In these cases, of course MS-LINK cannot include line numbers.)

▶

SWITCH	MEANING
/MAP (or /M)	MS-LINK lists all public (global) symbols defined in the output modules. If /MAP is not given, MSLINK will list only errors (including undefined globals). The symbols are listed alphabetically. For each symbol, MS-LINK lists its value and its segment: offset location in the run file. The symbols are listed at the end of the list file.
/NODEFAULTLIBRARY SEARCH (or /N)	MS-LINK does not automatically search the default library to resolve external references. For example, linking Pascal object modules with the /N switch stops MS-LINK automatically searching the file PASCAL.LIB.
/PAUSE (or /P)	MS-LINK pauses in the link session when the switch is encountered. Normally, MS-LINK performs the linking session without stopping from beginning to end. /PAUSE allows the user to swap diskettes before MS-LINK outputs the run (.EXE) file. When MS-LINK encounters the /PAUSE switch, it displays the message: **About to generate .EXE file** **Change disks - hit any key** MS-LINK resumes processing when the user presses any key. DO NOT SWAP A DISKETTE THAT IS TO RECEIVE A LIST FILE, OR A DISK USED FOR A TEMPORARY (VM.TMP) FILE.

▶

SWITCH	MEANING
/STACK:size (or /S)	The size of the stack provided for the load module by the assembler or compiler is overridden. The stack size becomes that specified in the 'size' parameter, which must follow the switch name and a colon. If a value from 1 to 511 is entered, MS-LINK uses 512. At least one object (input) module must contain a stack allocation statement. If not, MS-LINK will return a WARNING: NO STACK SEGMENT error message.

Characteristics

After any of these responses, before pressing **ENTER**, you may enter a comma (,) followed by the answer to what would be the next prompt, without having to wait for that prompt.

If you conclude any response with a semicolon (;) the remaining responses are all assumed to be the default. Linking begins immediately with no further prompting.

Use the plus sign (+) not only to separate lists of object files and libraries but to extend these lists, where necessary, onto more than one line. Enter the plus sign followed by **ENTER** at the end of a physical line. This repeats the object file or library prompt, and enables you to continue the logical line with further file names.

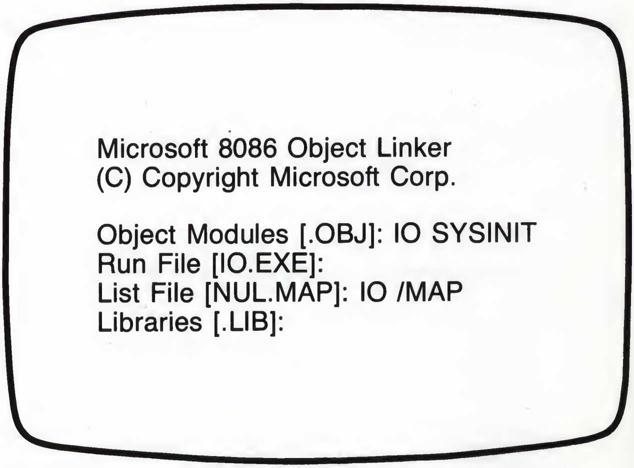
Example

This sample shows you a typical dialog for an MS-LINK session.

In response to the MS-DOS prompt, enter:

```
LINK
```

The system displays the following messages, prompts and your responses:



```
Microsoft 8086 Object Linker
(C) Copyright Microsoft Corp.

Object Modules [.OBJ]: IO SYSINIT
Run File [IO.EXE]:
List File [NUL.MAP]: IO /MAP
Libraries [.LIB]:
```

Fig. 9-1 LINK Example

Notes:

1. By specifying /MAP, you get both an alphabetic listing and a chronological listing of public symbols.
2. By responding PRN to the List File: prompt, you can redirect your output to the printer.
3. By specifying the /LINE switch, MS-LINK gives you a listing of all line numbers for all modules. (Note that the /LINE switch can generate a large volume of output).
4. By pressing **ENTER** in response to the Libraries: prompt, an automatic library search is performed.

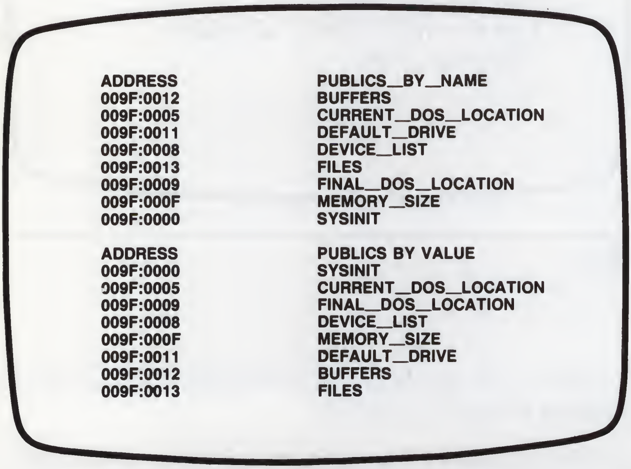
Once MS-LINK locates all libraries, the linker map displays a list of segments in the order of their appearance within the load module. The list might look like this:

```
Start     Stop      Length    Name
00000H    009ECH    09EDH     CODE
009FOH    01166H    0777H     SYSINITSEG
```

The information in the Start and Stop columns shows the 20-bit hex address of each segment relative to location zero. Location zero is the beginning of the load module.

The addresses displayed are not the absolute addresses where these segments are loaded.

Because the /MAP switch was used, MS-LINK displays the public symbols by name and value. For example:



```
ADDRESS              PUBLICS__BY__NAME
009F:0012            BUFFERS
009F:0005            CURRENT__DOS__LOCATION
009F:0011            DEFAULT__DRIVE
009F:0008            DEVICE__LIST
009F:0013            FILES
009F:0009            FINAL__DOS__LOCATION
009F:000F            MEMORY__SIZE
009F:0000            SYSINIT

ADDRESS              PUBLICS BY VALUE
009F:0000            SYSINIT
009F:0005            CURRENT__DOS__LOCATION
009F:0009            FINAL__DOS__LOCATION
009F:0008            DEVICE__LIST
009F:000F            MEMORY__SIZE
009F:0011            DEFAULT__DRIVE
009F:0012            BUFFERS
009F:0013            FILES
```

Fig. 9-2 /MAP Sample Display

COMMAND LINE ENTRY

With command line entry you enter the MS-LINK command along with its parameters, without waiting to be prompted.

You must separate each complete parameter entry from the next with a comma (,). Apart from this, what you enter is the same as with interactive entry. The command syntax is therefore the following:

[*d*:][*path*] **LINK** *objfile*[+*objfile*]...[*switch*]...[,[*runfile*][*switch*]...[,[*listfile*][*switch*]...[,[*libfile*[+*libfile*]...][*switch*]...]]]

Where

SYNTAX ELEMENT	MEANING
d:	Specifies the drive where LINK is to be found.
path	Specifies the directory where LINK is to be found.
objfile	An object module to be linked.
runfile	The run (executable) file to be created.
listfile	The list file to be output.
libfile	A library file to be searched.
switch	A switch to be applied.

Note

[*d*:][*path*] may precede any filename.

See the section ''Interactive Entry'' for full descriptions of these parameters.

Characteristics

Use the plus sign (+) as with interactive entry not only to separate lists of object files and library files but to extend those lists, where necessary, onto more than one line.

You may enter switches after any one of the four parameter entries (that is, before any of the commas or the final ENTER).

To accept the default parameter for a syntax element, enter a second comma with no space between the two commas. To accept a further default enter a third comma and so on. Remember that you must make an entry for the object file parameter.

If you enter a semicolon (;) at any time, the unspecified parameters all assume default values. Linking begins immediately.

If you enter an incomplete list of parameters and no semicolon is used, MS-LINK prompts you for the next remaining entry. See ''Interactive Entry'' for a full list of MS-LINK prompts.

Examples

IF you enter...	THEN...
LINK FUN+TEXT+TABLE+CARE/P/M,,FUNLIST,COBLIB.LIB	MS-LINK is loaded, then the object modules FUN.OBJ, TEXT.OBJ, TABLE.OBJ and CARE.OBJ are loaded. MS-LINK then pauses (because of the /P switch). The object modules are linked when you press any key, and a global symbol map is produced (because of the /M switch). The default run file FUN.EXE is created, along with a list file called FUNLIST. MAP. The library file COBLIB. LIB is searched for external references.
LINK FUN,,	MS-LINK is loaded, then the object module FUN.OBJ is loaded. The default run file FUN.EXE is created. MS-LINK then prompts for a list file, offering the file FUN.MAP as default.

AUTOMATIC RESPONSE FILE ENTRY

With automatic response file entry you enter the name of a file that already contains the answers to some or all of the MS-LINK parameter prompts. Precede this file name by the symbol @. The command syntax is therefore the following:

d:[*path*] **LINK** *@filespec*

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where LINK is to be found.
path	Specifies the directory where LINK is to be found.
filespec	The file specification of the file that contains responses to the MS-LINK prompts. Use of file extensions is optional; there is no default extension.

Characteristics

Automatic response files can contain several lines of text, each corresponding to an MS-LINK prompt. Responses must be in the same order as with interactive input.

Press **ENTER** to indicate the conclusion of each response and the beginning of the next. Type a plus sign (+) followed by **ENTER** to continue a response to the object module or libraries prompt on a new line. See the section ''Interactive Entry'' for full details of each MS-LINK prompt.

Use switches, commas and colons in an automatic response file just as you would in interactive entry.

You can enter the name of more than one automatic response file on the command line, and combine response file names with additional parameters. The combined series of resulting parameters must be a valid sequence of MS-LINK prompts.

When the MS-LINK session begins, each prompt is displayed in order with the responses from the response file. If the response file does not contain answers for all the prompts, (in the form of file names, the semicolon command character or carriage returns), MS-LINK displays the prompt which does not have a response, then waits for you to enter a legal response. When a legal response has been entered, MS-LINK continues the link session.

Example

IF you enter...	THEN...
FUN TEXT TABLE CARE /PAUSE/MAP FUNLIST COBLIB.LIB	this response file tells MS-LINK to load the four object modules named FUN, TEXT, TABLE, and CARE. MS-LINK pauses before producing a public symbol map to permit you to swap diskettes. When you press any key, the output files are named FUN.EXE and FUNLIST.MAP. MS-LINK searches the library file COBLIB.LIB.

10. THE DEBUGGER

ABOUT THIS CHAPTER

This chapter describes the DEBUG utility.

CONTENTS

INTRODUCTION **10-1**

HOW TO INVOKE THE DEBUG PROGRAM **10-1**

DEBUG **10-1**

DEBUGGING COMMANDS **10-4**

COMMAND PARAMETERS **10-4**

A (ASSEMBLE) **10-7**

C (COMPARE) **10-10**

D (DUMP) **10-11**

E (ENTER) **10-14**

F (FILL) **10-17**

G (GO) **10-18**

H (HEX) **10-21**

I (INPUT) **10-23**

L (LOAD) **10-24**

M (MOVE) **10-27**

N (NAME) **10-29**

O (OUTPUT) **10-31**

P (PROCEED) **10-32**

Q (QUIT) **10-33**

R (REGISTER) **10-34**

S (SEARCH) **10-37**

T (TRACE) **10-39**

U (UNASSEMBLE) **10-40**

W (WRITE) **10-42**

INTRODUCTION

The DEBUG utility is a program that provides a controlled testing environment for binary and executable object files. It eliminates the need to re-assemble a program to see if a problem has been corrected by a minor change. Moreover, it enables you to change the contents of a file or CPU register, then to immediately re-execute a program to check the validity of the changes.

HOW TO INVOKE THE DEBUG PROGRAM

The DEBUG program is invoked as follows:

[*d*:][*path*] **DEBUG** [*filespec*[, *arglist*]]

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where DEBUG is to be found.
path	Specifies the directory where DEBUG is to be found.
filespec	The specifier of the program file to be debugged.

SYNTAX ELEMENT	MEANING
arglist	A list of file name parameters and switches separated by commas. These will be passed to the program specified by the *filespec* parameter. Thus, when the program is loaded into memory, it is loaded as if it had been invoked with the command: *filespec arglist* That is, *filespec* indicates the file to be debugged, and *arglist* is the rest of the command line that is used when the file is invoked and loaded into memory.

Characteristics

On entering the DEBUG environment DEBUG responds with the hyphen (-) prompt and underline (_) cursor. You are then free to enter any DEBUG command.

If you enter DEBUG without parameters, since no file name has been specified, current memory, disk blocks, or disk files can be worked on using other debugging commands.

If you include the *filespec* in the command line then the specified file is loaded into memory starting at location 100 (hexadecimal). However, if the file has an EXE extension, then it is relocated to the address specified in the header of the file. Moreover, if the file has the HEX extension, then the file is loaded beginning at the address specified in the HEX file.

Examples

IF you enter...	THEN...
DEBUG	the DEBUG environment is entered, but without loading a file.
DEBUG b:myprog	the DEBUG environment is entered and the file named ''myprog'' is loaded into memory from drive B.
DEBUG progs\dev\ myrtn, prog1,prog2	the DEBUG environment is entered and the file named ''myrtn'' is loaded into memory from the sub-directory ''dev'' of directory ''progs''. The loaded file also takes two file name parameters: ''prog1'' and ''prog2''.

Remarks

When you invoke DEBUG, it sets up a program header at offset 0 in the program work area. You can overwrite this area if you enter DEBUG without parameters. Moreover, if you are debugging a file with a COM or EXE extension you must not tamper with the program header below location 5CH, or DEBUG will terminate.

Do not restart a program after the ''Program terminated normally'' message is displayed. You must reload the program with the N and L commands for it to run properly.

DEBUGGING COMMANDS

This section describes the DEBUG commands in alphabetical order for ease of reference. Each such command description summarizes the purpose of the command, defines the command syntax and explains each syntax element. This is followed, for each command, by a detailed account of the command characteristics and some working examples.

Remarks

1. Commands can be entered in either upper or lower case.
2. Command keywords and command parameters can be separated from each other by spaces or commas for readability but need not be, except where two hexadecimal numbers are entered as parameters, in which case they must be separated by a comma or space. For brevity, the syntax of this chapter will always indicate comma where separation is obligatory, but note that a space can alternatively be used.
3. Commands only become effective after pressing **ENTER**
4. If you make a syntax error when entering a command the message ''^Error'' will be displayed. You must re-enter the command using the correct syntax.

COMMAND PARAMETERS

The following DEBUG command parameters require special definition.

PARAMETER	DEFINITION
address	A hexadecimal value in one of the following formats: • A segment register designation and a hexadecimal offset separated from each other by a colon. For example: DS:0300 • A hexadecimal segment and offset separated from each other by a colon. For example: 9D0:0100 • A hexadecimal offset value. For example: 200 The DEBUG command will append a default segment value from either the DS or CS registers, depending on the command.
byte	A one or two character hexadecimal value.
drive	0, 1 or 2 depending on whether you wish to select drive A, drive B or drive C, respectively.

▶

PARAMETER	DEFINITION
range	A range of addresses specified as: EITHER: address L value where address specifies the start of the range and value specifies the length of the range. For example: DS:300L30 indicates a range of 48 locations starting at address 300 in the sector indicated by the DS register. The specified range cannot be greater than 1000 (hexadecimal). To specify this value enter 0000 (or 0) as the value parameter. OR: address,address where the two addresses indicate the limits of the range. Note that space may be used instead of comma.
value	A 1 to 4 character hexadecimal value.

A (ASSEMBLE)

Assembles assembler mnemonics directly into memory.

A [*address*]

Where

SYNTAX ELEMENT	MEANING
address	The start address at which the subsequently entered line of mnemonics is to be assembled. If this parameter is omitted, location 100 is assumed.

Characteristics

After you enter the Assemble command, DEBUG displays the specified *address* followed by the cursor. You may then enter a line of assembler mnemonics. On terminating the line with **ENTER** the line will be assembled into memory starting at the specified location. The address of the byte subsequent to the assembled code will be displayed on the next line along with the cursor to enable you enter the next line of code. If, instead of a line of assembler mnemonics, you simply press **ENTER** , the Assemble command terminates and the DEBUG prompt re-appears.

All numeric values are hexadecimal and must be entered as 1 to 4 characters. Prefix mnemonics must be specified in front of the opcode to which they refer. You may also enter them on a separate line.

The segment override mnemonics are CS:, DS:, ES: and SS:. The mnemonic for the far return is RETF. String manipulation mnemonics must explicitly state the string size. For example, use MOVSW to move word strings and MOVSB to move byte strings.

The Assemble command will automatically assemble short, near or far jumps and calls, depending on byte displacement with respect to the destination address. These may be overridden with the NEAR or FAR prefix. For example:

```
0100:0500 JMP 502          ;a two-byte short jump
0100:0502 JMP NEAR 505     ;a three-byte near jump
0100:505  JMP FAR 50A      ;a five-byte far jump
```

The NEAR prefix may be abbreviated to NE, but the FAR prefix cannot be abbreviated.

DEBUG cannot tell whether some operands refer to a word memory location or to a byte memory location. In this case the data type must be explicitly stated with the prefix ''WORD PTR'' or ''BYTE PTR''. Acceptable abbreviations are ''WO'' and ''BY''. For example:

```
NEG BYTE PTR [128]
DEC WO [SI]
```

A simple operand is a literal. Whereas operands enclosed within square brackets refer to memory. For example:

```
MOV AX,21                  ;Load AX with 21H
MOV AX,[21]                ;Load AX with the contents of location
                           ;21H
```

Two popular pseudo-instructions are available with the Assemble command. The DB opcode will assemble word values directly into memory. For example:

```
DB 1,2,3,4,''THIS IS AN EXAMPLE''
DB 'THIS IS A QUOTE: '''
DB ''THIS IS A QUOTE'''

DW 1000,2000,3000,''BACH''
```

The Assemble command supports all forms of register indirect commands. For example:

```
ADD BX,34[BP+2].[SI-1]
POP [BP+DI]
PUSH [SI]
```

All opcode synonyms are also supported. For example:

```
LOOPZ 100
LOOPE 100
JA 200
JNBE 200
```

Example

IF you enter...	THEN...
A200	DEBUG displays **09AC:0200 _**
MOV AX,[21]	the assembler mnemonics are assembled starting at location 200. The byte location subsequent to the assembled code is then displayed thus **09AC:0203 _**
ENTER	the Assemble command terminates and the DEBUG prompt re-appears.

C (COMPARE)

Compares the contents of two areas of memory.

C *range* , *address*

Where

SYNTAX ELEMENT	MEANING
range	The range of addresses defining the first area to be compared. If no segment is specified then the segment specified in the D register is assumed.
address	The start of the area to be compared with the area specified by the range parameter.

Characteristics

The Compare command compares the area of memory specified by the *range* parameter with an area of the same size starting at the location specified by the *address* parameter.

If the contents of the two areas are identical nothing is displayed. If there are differences, then the differences are displayed in the form:

```
address1 contents1 contents2 address2
```

where: address1 indicates the address in the first area and contents1 its contents; and address2 indicates the corresponding address in the second area and contents2 its contents.

Example

IF you enter...	THEN...
C100, 1FF, 300 or C100L100, 300	the area of memory from 100 to 1FF is compared with the area of memory from 300 to 3FF.

D (DUMP)

Displays an area of memory.

D [*range* \| *address*]

Where

SYNTAX ELEMENT	MEANING
range	The range of addresses whose contents are to be displayed. If you enter only an offset, then the segment specified in the DS register is assumed.
address	The address from which the display is to start. The contents of this address and the subsequent 127 locations are displayed.

Characteristics

If D is specified without parameters then the 128 bytes following the last address to be displayed are displayed. If no location has yet been accessed then the dump will start from location DS:100.

If D and the *range* parameter are specified then the contents of that range of addresses are displayed. If this takes more than 24 screen lines the display is scrolled until the contents of the final address in the range are displayed on line 24.

If D and a single address are specified, then the contents of the 128 locations starting from the specified address are displayed.

The dump is displayed in two portions:

- A hexadecimal dump, where each byte is represented by its hexadecimal value.
- An ASCII dump, where the equivalent ASCII character for the byte is displayed. If there is no corresponding printable ASCII character then the dump displays a period (.).

Each line of the dump begins with an address which is followed by the hexadecimal contents of the 16 bytes starting from the addressed location. The eighth and ninth bytes are separated by a hyphen (-). The right-hand columns display the equivalent ASCII values. Each line of the display, except possibly the first, begins on a 16 byte boundary.

Examples

IF you enter...	THEN...
D 100,110 or D100L11	a hexadecimal and ASCII dump of lines 100 to to 110 (hexadecimal), inclusive, are displayed.
D	a hexadecimal and ASCII dump of the 128 bytes starting from location 111 (hexadecimal) is displayed.
D200	a hexadecimal and ASCII dump of the 128 bytes starting from location 200 (hexadecimal) is displayed.

E (ENTER)

Replaces the contents of memory locations at the byte addresses specified.

E *address*[, *bytevalue*[, *bytevalue*]...]

Where

SYNTAX MEANING	MEANING
address	The address of the location whose value is to be replaced; or the address of the first of a succession of locations whose contents are to be replaced. If only an offset is specified then the segment indicated by the DS register is assumed.
bytevalue	The hexadecimal byte value that is to replace the contents of the specified address. The first *bytevalue* parameter will replace the contents of the location specified by the address parameter. A second *bytevalue* will replace the contents of the location following that specified by the address parameter, and so on.

Characteristics

If the command is entered without the *bytevalue* list, then DEBUG displays the specified *address* and its contents. The Enter command then waits for you to perform one of the following:

- Replace the displayed byte value by entering another value. You simply enter the new value after the current value. If you enter an illegal value, or if you type more than two digits then the illegal or extra character is not echoed.

- Advance to the next byte by pressing **SPACE** . To change the value of this byte simply enter the value as described above. If you advance beyond an eight-byte boundary, DEBUG starts a new display line with the address displayed at the start of the line. To advance to the next byte without changing the current byte simply press **SPACE** again.

- Return to the previous byte by entering hyphen (-). On doing so DEBUG starts a new display line with the address of the byte you have returned to and its contents. You can then change the contents of this location as described above. To move back one byte further without changing this value simply enter hyphen again, and another new display line will be generated.

- Terminate the Enter command by pressing **ENTER** . This key may be pressed in any byte position.

If you specify *bytevalues* in the command line then the first of these *bytevalues* will replace the contents of the location specified by the *address* parameter. Subsequent entries in the list of *bytevalues* will replace subsequent bytes in memory.

Examples

IF you enter...	THEN...
E100	DEBUG displays something like: **058D:0100 CD.__**
26	the value of location 100 is changed to 26 and DEBUG displays: **058D:0100 CD.26__**
SPACE	the next byte (location 101) is displayed: **058D:0100 CD.26 20.__**
SPACE	the next byte (location 102) is displayed: **058D:100 CD.26 20. 00.__**
-	the previous byte (location 101) is displayed on the next line: **058D:0100 CD.26 20. 00.** **058D:0101 20.__**
30	the contents of location 101 are changed to 30 and the Enter command is terminated: **058D:0100 CD.26. 20. 00.** **058D:0101 20.30** 〉__
E200,26,0A,19,23	the contents of byte locations 200, 201, 202 and 203 are changed to 26, 0A, 19 and 23, respectively

F (FILL)

Fills an area of memory with specified byte values.

F *range* , *bytevalue*[, *bytevalue*...]

Where

SYNTAX ELEMENT	MEANING
range	The range of addresses whose contents are to be overwritten with the specified byte values. If only the offset is specified then the segment indicated by the DS register is assumed.
bytevalue	A two digit hexadecimal value that is to overwrite the contents of the specified address.

Characteristics

If the specified *range* contains more bytes than the list of *bytevalues* , then the list of byte values is repeated until the specified range is filled.

If the list of *bytevalues* is longer than the specified *range* then the extra *bytevalues* are ignored.

Example

IF you enter...	THEN...
F04BA:100L100,42,45, 48,37,20	DEBUG fills memory locations 04BA:100 to 04BA:1FF with the byte values specified. The five values are repeated until all 256 locations are filled.

G (GO)

Executes the program currently in memory, optionally halting at a specified breakpoint and displaying information about the system/program environment.

G [= *address*][, *address*]...

Where

SYNTAX ELEMENT	MEANING
=*address*	The address in memory at which program execution is to start." = " must be entered to distinguish a start address from a breakpoint address.

▶

SYNTAX ELEMENT	MEANING
address	The breakpoint address at which program execution is to halt and the register and flag states to be displayed along with the next instruction to be executed.

Characteristics

If you enter G without parameters then the program currently in memory is executed starting from the address specified by the CS and IP registers.

If you specify the =*address* parameter, then the contents of the CS and IP registers are changed to those specified by the =*address* parameter, and the program in memory is executed, starting from that point.

If you specify one or more breakpoint addresses then program execution stops at the first such address encountered and displays the contents of the registers, the state of the flags and the next instruction to be executed (see the Register command for a description of the display).

You may specify up to ten breakpoint addresses, in any order. If your program has many paths you can use this feature to ensure that your program halts, whichever path it takes.

If you enter more than ten breakpoints DEBUG will display:

BP Error

Before executing the program the Go command replaces the contents of the breakpoint locations with an interrupt instruction (hexadecimal CC). When program execution halts at such a location DEBUG restores the original values of all the specified breakpoint locations. However, if the program terminates normally (that is, not at a specified breakpoint), then the breakpoint values are not restored.

Each breakpoint address that you specify must point to the first byte of an 8086 instruction, otherwise unpredictable results will occur.

The user stack pointer must have six bytes available for this command, otherwise unpredictable results will occur.

Example

IF you enter...	THEN...
G = 200,1AF,141	the program currently in memory is executed starting from location 200. Assuming location 141 is encountered before 1AF, then the program halts at location 141 and the register and flag values are displayed along with the next instruction to be executed. If neither breakpoint location is encountered, then the program terminates normally.
G	if, in the previous example, the progrm halted at location 141, then program execution continues from that address. If program execution terminated normally in the previous example, then program execution again starts at location 200.

H (HEX)

Calculates and displays the sum and the difference of two hexadecimal values.

H *value-a* **,** *value-b*

Where

SYNTAX ELEMENT	MEANING
value-a	The first of two hexadecimal values.
value-b	The hexadecimal value that is to be added to or subtracted from *value-a*.

Characteristics

The hexadecimal values may be up to four characters long.

The Hex command displays two four digit values:

- The first is the result of adding *value-b* to *value-a*
- The second is the result of subtracting *value-b* from *value-a*

Example

IF you enter...	THEN...
H19F,10A	DEBUG displays: 02A9 0095
HFFFF,2	DEBUG displays: 0001 FFFD

I (INPUT)

Inputs and displays (in hexadecimal) one byte from the specified port.

I *value*

Where

SYNTAX ELEMENT	MEANING
value	The hex address of the port from which the byte is to be input.

Characteristics

The port address can be up to 16 bits.

Example

IF you enter...	THEN...
I2F8	the byte at the addressed port is input and displayed.

L (LOAD)

Loads a file or absolute disk sectors into memory.

L [*address*[, *drive* , *sector-a* , *sectors*]]

Where

SYNTAX ELEMENT	MEANING
address	The address in memory at which the file or specified sectors, is to be loaded. If only an offset is entered then the segment indicated by the CS register is assumed. Sectors cannot be loaded across segments.
drive	The drive from which disk sectors are to be loaded. For drive A you must enter 0, for drive B you must enter 1, or for drive C you must enter 2.
sector-a	The first of a range of sectors to be loaded from the disk specified by the drive parameter.
sectors	The number of sectors to be loaded. The maximum number of sectors that can be specified is 80 Hexadecimal.

Characteristics

If all parameters are specified then DEBUG loads sectors of information from disk into memory. The first such sector to be loaded is that specified by the *sector-a* parameter. It is loaded at the memory location specified by the *address* parameter. This sector is the first of a continuous range of sectors to be loaded, the number of which is specified by the *sectors* parameter.

If you enter L without parameters, or with just the *address* parameter, the file whose file control block is correctly formatted at location CS:5C is loaded into memory. The file control block at CS:5C is set either to the *filespec* specified when the DEBUG command was invoked, or to the *filespec* specified by the most recent subsequent Name command.

If L is entered alone, then the file is loaded at location CS:100. If you specify L and the *address* parameter, the file is loaded at the specified address. In either case DEBUG sets the BX:CX registers to the number of bytes loaded.

If the file has an .EXE extension, then it is relocated to the load address specified in the loader of the .EXE file. That is, the *address* parameter to the Load command is ignored. Note that the header itself is stripped off the .EXE file before the file is loaded into memory. Thus the size of the .EXE file on disk will differ from its size in memory.

If the file is a .HEX file, then entering the Load command with no parameters causes the file to be loaded starting at the address specified within the .HEX file. If the *address* parameter, however, is specified then loading starts at the address which is the sum of the address specified and the address in the .HEX file.

Examples

The following examples assume the system to be initially in MS-DOS.

IF you enter...	THEN...
debug Nb:file.com L	the debugger is entered and the subsequent Name command sets the file control block at CS:5C to identify file "file.com" on the diskette inserted in drive B. The Load command then loads this file into memory starting at CS:100 (the default address).
debug b:file.com L300	file.com is loaded into memory at location CS: 100 by the DEBUG command. It is then relocated to CS:300 by the Load command.
debug L500,1,OF,6D	109 sectors are loaded into memory from drive B starting from sector OF. They are placed in memory starting at location CS:500.

M (MOVE)

Moves the contents of a specified range of memory addresses to the locations starting at a specified address.

M *range* , *address*

Where

SYNTAX ELEMENT	MEANING
range	The area of memory whose contents are to be moved. If you only entered an offset then the segment indicated in the DS register is assumed.
address	The start of the destination area. If you only entered an offset then the segment indicated by the DS register is assumed.

Characteristics

If the source and destination areas overlap the move is performed without loss of data.

The contents of the source area are not changed by the move, unless the destination area overlaps it.

If you specify an address as the end of the range you must only enter the offset. The segment specified, or defaulted to, in the start address of the range is assumed.

Example

IF you enter...	THEN...
MCS:100,110,CS:500 or MCS:100L11,CS:500	the 11 bytes starting at location CS:100 through to 110 are copied to the 11 bytes starting at location CS:500.

N (NAME)

Provides file names for the Load and Write commands or file name parameters for the program to be debugged.

N *filespec*[, *filespec*]...

Where

SYNTAX ELEMENT	MEANING
filespec	The file specifier of a file to be loaded into memory, written to diskette, or used as a file name parameter to the file currently in memory.

Characteristics

The name command can be used to provide:

- The name of the disk file to be loaded into memory by a subsequent Load command.
- The name to be assigned to the file currently in memory when the file is subsequently written to disk.
- File name parameters to the file in memory to be debugged.

The first case enables you to specify the file you wish to debug after entering the DEBUG environment. That is, you can enter DEBUG without specifying parameters, then use the Name command to name the disk file you wish to debug, then load the file into memory using the Load command. This has the same effect as entering the file name as the first parameter to the DEBUG command. In either case the file control block for the filé to be debugged is set up at location CS:5C and the file is loaded.

In the second case the file is already in memory and the Name command sets up the file control block for the specified file name at location CS:5C. When a Write command is subsequently entered the file in memory is written to disk with the file name whose file control block is set up at location CS:5C.

In the third case the Name command provides file name parameters for the program currently in memory. Whatever file control block was set at CS:5C is replaced by that of the first such parameter. If a second file parameter is specified then its file control block is set up at location CS:6C. Only two file control blocks are set up although additional file name parameters may be included if required. All the file names specified are placed in a save area at CS:81, with CS:80 containing a character count. Parameters specified in this way are analogous to file names specified in the argument list to the DEBUG command.

Examples

IF you enter...	THEN...
DEBUG Nb:file.com L	the system enters the DEBUG environment and the file named file.COM resident on drive B has its file control block set up at location CS:5C. The Load command subsequently loads this file into memory. This sequence has the same effect as entering "DEBUG b:file.com".

IF you enter...	THEN...
Nb:newfile.com W	the file control block is set up at location CS:5C for the file specifier ''b:newfile.com''. The subsequent Write command writes the file currently in memory to drive B and names the file ''newfile.com''.
DEBUG b:file1.com Nfile2.dat,file3. dat G	the DEBUG command loads the file named file1.com from drive B to be debugged. The Name command sets up two file control blocks at locations CS:5C and CS:6C for the file specifiers b:file2. dat and b:file3.dat, respectively. These files then become parameters to file1.com when the subsequent Go command executes file1.com.

O(OUTPUT)

Sends a specified byte to an output port.

O *value* , *byte*

Where

SYNTAX ELEMENT	MEANING
value	The address of the output port. It must be specified in hexadecimal and can be up to 16 bits.
byte	A two-digit hexadecimal value to be sent to the specified port.

Example

IF you enter...	THEN...
01E8, 27	the byte value 27 Hex is output to the port IE8.

P (PROCEED)

Proceed past a CALL or INT instruction.

P

Characteristics

The Proceed command is similar to the Trace command, but it accepts no parameters. Its function is to execute, without tracing, all CALL and INT instructions. In addition it displays repeated instructions only once and traces loops (terminated by LOOP XX) only on the first pass.

Q(QUIT)

Terminates the DEBUG program.

Q

Characteristics

The Quit command terminates the debugger without saving the file you are working on. Control is returned to MS-DOS command mode.

Example

IF you enter...	THEN...
Q	the DEBUG program terminates and returns you to MS-DOS command mode.

R(REGISTER)

Displays the hexadecimal contents of the registers and flag settings, or displays the contents of a specified register with the option to change that value, or displays the flag settings with the option of reversing any number of those settings.

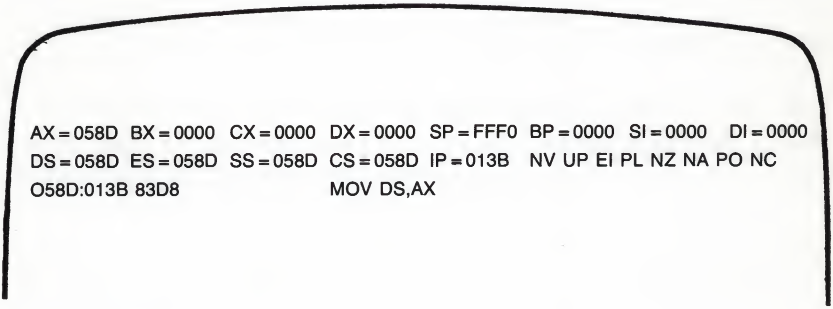
R [*register-name* | **F**]

Where

SYNTAX ELEMENT	MEANING
register-name	Any valid register name whose contents are to be examined and optionally changed. This may be one of: AX DX SI ES IP BX SP DI SS PC CX BP DS CS Note: IP and PC both refer to the Instruction Pointer.
F	The flag settings are to be displayed and, optionally, changed.

Characteristics

If you enter R without parameters, then the contents of all registers are displayed along with the flag settings and the next instruction to be executed. For example:



```
AX=058D  BX=0000  CX=0000  DX=0000  SP=FFF0  BP=0000  SI=0000   DI=0000
DS=058D  ES=058D  SS=058D  CS=058D  IP=013B   NV UP EI PL NZ NA PO NC
O58D:013B 83D8                     MOV DS,AX
```

Fig. 10-1 Sample R Display

If you enter R with a *register-name*, then DEBUG displays the contents of that register. The command then waits for you to do one of the following:

- Press **ENTER** to terminate the Register command without changing the value of the displayed register.

- Change the value of the register by entering the four-digit hexadecimal value then terminate the Register command by pressing **ENTER**.

The valid flag values are shown in the following table:

FLAG NAME	SET	CLEAR
Overflow	OV (yes)	NV (no)
Direction	DN (decrement)	UP (increment)
Interrupt	EI (enabled)	DI (disabled)
Sign	NG (negative)	PL (plus)
Zero	ZR (yes)	NZ (no)
Auxiliary carry	AC (yes)	NA (no)
Parity	PE (even)	PO (odd)
Carry	CY (yes)	NC (no)

If you enter RF, then the current flag settings are displayed. You can then either:

- Press **ENTER** to terminate the Register command without changing the flag values, or
- Change the setting of one or more flags by entering the alternate value of the appropriate flags. The new values may be entered in any order, with or without delimiters.

Examples

IF you enter...	THEN...
R	DEBUG displays the contents of all registers, flag settings and the next instruction to be executed.
R IP	DEBUG displays the contents of the Instruction Pointer. For example: **IP 0139** **:__**
013B	the contents of the Instruction Pointer are changed to 013B.
RF	DEBUG displays the flag settings. For example: **NV UP EI PL NZ NA PO NC-__**
PE ZR DI NG	The Parity flag is set to even (PE), the Zero flag is set (ZR), the Interrupt flag is cleared (DI), and the Sign flag is set (NG).
RF	DEBUG displays the new state of the flags: **NV UP DI NG ZR NA PE NC-__**

S (SEARCH)

Searches a specified range for a list of bytes.

S *range* , *list*

Where

SYNTAX ELEMENT	MEANING
range	The range of addresses within which the search is to be made. If you only enter the offset then the segment indicated by the DS register is assumed.
list	The list of bytes to be searched for. Bytes in the list must be separated by a space or a comma.

Characteristics

For each occurrence of the list of bytes within the specified range, DEBUG returns the address of the first byte. If no address is returned, no match was found.

Example

IF you enter...	THEN...
S100L100,20 or S100,1FF,20	DEBUG displays the address of every occurrence of byte value 20 in the address range 100 to 1FF, inclusive. For example: 058D: 010C 058D: 0110 058D: 0115 058D: 0118 058D: 0120 058D: 0128 058D: 0125

T (TRACE)

Executes one or more instructions and displays the register contents, flag settings and the next instruction to be executed.

T [= *address*][, *value*]

Where

SYNTAX ELEMENT	MEANING
=*address*	DEBUG is to commence execution at this address.
value	The number of instructions to be executed.

Characteristics

If the =*address* parameter is not specified then execution begins at CS:IP.

If the *value* parameter is not specified then only one instruction is executed.

The display generated is of the same format as that of the Register command (without parameters).

Examples

IF you enter...	THEN...
T = 200,5	five instructions, starting with the one at location 200 are executed, and the register and flag values following each instruction are displayed along with the next instruction to be executed.
T	the instruction pointed to by CS:IP is executed and the register and flag contents are displayed along with the next instruction to be executed.

U (UNASSEMBLE)

Disassembles strings of bytes in memory and displays them as assembler-like statements along with their corresponding addresses.

U [*range*|*address*]

Where

SYNTAX ELEMENT	MEANING
range	The range of addresses whose byte values are to be disassembled. If you do not specify the segment then the segment indicated by the CS register is assumed.
address	The start of a 32 byte area of memory to be disassembled. If you only enter an offset then the segment indicated by the CS register is assumed.

Characteristics

If neither the *range* nor *address* parameter is specified, then 32 bytes are disassembled starting at location CS:IP (16 bytes are disassembled in 40 column screen mode).

The number of bytes disassembled may be slightly more than the number you specified. This is because instructions are not always the same length and the final address in a range will not always contain the last byte of an instruction.

The first address of a range, or the *address* parameter, must always refer to the first byte of an assembler instruction, otherwise results will be unpredictable.

Example

IF you enter...	THEN...
U058D:204L8	eight bytes starting at location 058D:204 are disassembled and the result displayed: **058D:0204 8D16DFOD LEA DX,[0DDF]** **058D:0208 42 INC DX** **058D:0209 03D0 ADD DX,AX** **058D:020B 8916E50B MOV [0BE5],DX**

W (WRITE)

Writes the file being debugged to disk.

W [*address*[, *drive* , *sector-a* , *sectors*]]

Where

SYNTAX ELEMENT	MEANING
address	The start address of the code in memory that is to be written to disk. If you enter only an offset then the segment indicated in the CS register is assumed.
drive	The drive containing the specified blocks to which code in memory is to be written. For drive A you must enter 0, for drive B you must enter 1, or for drive C you must enter 2.
sector-a	The sector number on disk that is the first of a contiguous range of sectors to be overwritten with code from memory.
sectors	The number of disk sectors blocks to be overwritten with code from memory. The maximum number of sectors that can be specified is 80 Hexadecimal.

Characteristics

If you enter the Write command without parameters, then the file is written to disk starting from memory address CS:100. If you specify the *address* parameter then the file in memory, starting from the specified address, is written to disk.

In either case, before executing the Write command, BX:CX must be set to the number of bytes to be written. This value was set up correctly when the file was loaded (either by the Load command or the DDEBUG command itself). However, if, since loading the file, you have executed a Go or Trace command, then the value of BX:CX will have been changed. You must be sure this value is set up correctly.

When the Write command writes a file to disk it obtains the drive specifier and file name via the file control block set up at CS:5C. If no drive specifier is set up then the default is assumed. This file control block is set up either by the DEBUG command or a subsequent Name command. If it does not indicate the file specifier you require, you must set up this file control block using the Name command.

When the file is written to disk it overwrites the version currently on disk unless the specified file name does not exist, in which case a new file is created.

If all parameters are specified then the code in memory is written to the drive specified by the parameter. The data to be written starts at the memory location specified by the *address* parameter, and is written to the blocks on disk specified by the *sector-a* and *sectors* parameters. You must therefore be **extremely** careful to specify the required sectors, since information held there will be destroyed by this operation.

Examples

IF you enter...	THEN...
W	the file in memory, starting from location CS:100, is written to disk with the file specifier defined by the file control block set up at location CS:5C.
W200	the file in memory, starting from location CS:200, is written to disk with the file specifier defined by the file control block set up at location CS:5C.
W200,1,1F,20	20 Hex (32 decimal) sectors on drive B starting from location IF are overwritten with the data starting at memory location CS:200.

A. ASCII DISPLAY AND KEYBOARD CODE TABLES

ABOUT THIS APPENDIX

This appendix provides a table of ASCII codes and extended keyboard codes.

CONTENTS

BASIC ASCII CODE (ISO 7-bit code) A-1

EXTENDED ASCII CODE FOR THE DISPLAY (ISO 8-bit code) A-5

NATIONAL VARIATIONS IN EXTENDED ASCII CODE (ISO 8-bit code) A-8

EXTENDED KEYBOARD A-14

BASIC ASCII CODE (ISO 7-bit code)

This is a 7 bit code used for Information Interchange. The codes 0 to 31 and code 127 (all decimal numbers) are Control Codes. These Control Codes are used for Data Transmission, Device Control and Formatting Effects. Note that CTRL shows on the screen as ^.

DECIMAL	HEXADECIMAL	CHARACTER	CONTROL	MEANING
0	00	(BLANK)	NUL	Null
1	01	CTRL A	SOH	Start Of Heading
2	02	CTRL B	STX	Start of Text
3	03	CTRL C	ETX	End of Text
4	04	CTRL D	EOT	End of Transmission
5	05	CTRL E	ENQ	Enquiry
6	06	CTRL F	ACK	Acknowledge
7	07	CTRL G	BEL	Bell
8	08	CTRL H	BS	Backspace
9	09	CTRL I	HT	Horizontal Tabul.
10	0A	CTRL J	LF	Line Feed
11	0B	CTRL K	VT	Vertical Tabulation
12	0C	CTRL L	FF	Form Feed
13	0D	CTRL M	CR	Carriage Return
14	OE	CTRL N	SO	Shift-Out
15	OF	CTRL O	SI	Shift-In
16	10	CTRL P	DLE	Data Link Escape
17	11	CTRL Q	DC1	Device Control (1)
18	12	CTRL R	DC2	Device Control (2)
19	13	CTRL S	DC3	Device Control (3)
20	14	CTRL T	DC4	Device Control (4)
21	15	CTRL U	NAK	Negative Acknowl.
22	16	CTRL V	SYN	Synchronous Idle
23	17	CTRL W	ETB	End of Trans. Block
24	18	CTRL X	CAN	Cancel
25	19	CTRL Y	EM	End of Medium
26	1A	CTRL Z	SUB	Substitute Charac.
27	1B	CTRL [	ESC	Escape
28	1C	CTRL \	FS	File Separator
29	1D	CTRL]	GS	Group Separator
30	1E	CTRL ^	RS	Record Separator
31	1F	CTRL _	US	Unit Separator
32	20	(SPACE)	SP	Space

▶

DECIMAL	HEXADECIMAL	CHARACTER	CONTROL	MEANING
33	21	!		
34	22	’’		
35	23	#		
36	24	$		
37	25	%		
38	26	&		
39	27	’		
40	28	(		
41	29	)		
42	2A	*		
43	2B	+		
44	2C	,		
45	2D	-		
46	2E	.		
47	2F	/		
48	30	0		
49	31	1		
50	32	2		
51	33	3		
52	34	4		
53	35	5		
54	36	6		
55	37	7		
56	38	8		
57	39	9		
58	3A	:		
59	3B	;		
60	3C	<		
61	3D	=		
62	3E	>		
63	3F	?		
64	40	@		
65	41	A		
66	42	B		
67	43	C		
68	44	D		
69	45	E		
70	46	F		
71	47	G		

DECIMAL	HEXADECIMAL	CHARACTER	CONTROL	MEANING
72	48	H		
73	49	I		
74	4A	J		
75	4B	K		
76	4C	L		
77	4D	M		
78	4E	N		
79	4F	O		
80	50	P		
81	51	Q		
82	52	R		
83	53	S		
84	54	T		
85	55	U		
86	56	V		
87	57	W		
88	58	X		
89	59	Y		
90	5A	Z		
91	5B	[		
92	5C	\		
93	5D	]		
94	5E	^		
95	5F	_		
96	60	`		
97	61	a		
98	62	b		
99	63	c		
100	64	d		
101	65	e		
102	66	f		
103	67	g		
104	68	h		
105	69	i		
106	6A	j		
107	6B	k		
108	6C	l		
109	6D	m		
110	6E	n		

DECIMAL	HEXADECIMAL	CHARACTER	CONTROL	MEANING
111	6F	o		
112	70	p		
113	71	q		
114	72	r		
115	73	s		
116	74	t		
117	75	u		
118	76	v		
119	77	w		
120	78	x		
121	79	y		
122	7A	z		
123	7B	{		
124	7C	|		
125	7D	}		
126	7E	~		
127	7F	(DELETE)	DEL	Delete previous character

EXTENDED ASCII CODE FOR THE DISPLAY (ISO 8-bit code)

This table shows the 256 elements of the extended ASCII character set, together with their decimal and hexadecimal equivalents.

DEC:	HEX	CHARACTER	DEC:	HEX	CHARACTER	DEC:	HEX	CHARACTER	DEC:	HEX	CHARACTER
000	00	BLANK (NULL) (NUL)	016	10	► (DLE)	032	20	(SPACE) (SP)	048	30	0
001	01	☺ (SOH)	017	11	◄ (DC1)	033	21	!	049	31	1
002	02	☻ (STX)	018	12	↕ (DC2)	034	22	”	050	32	2
003	03	♥ (ETX)	019	13	‼ (DC3)	035	23	#	051	33	3
004	04	◆ (EOT)	020	14	¶ (DC4)	036	24	$	052	34	4
005	05	♣ (ENQ)	021	15	§ (NAK)	037	25	%	053	35	5
006	06	♠ (ACK)	022	16	▬ (SYN)	038	26	&	054	36	6
007	07	• (BEL)	023	17	↨ (ETB)	039	27	’	055	37	7
008	08	◘ (BS)	024	18	↑ (CAN)	040	28	(	056	38	8
009	09	○ (HT)	025	19	↓ (EM)	041	29	)	057	39	9
010	0A	◙ (LF)	026	1A	→ (SUB)	042	2A	*	058	3A	:
011	0B	♂ (VT)	027	1B	← (ESC)	043	2B	+	059	3B	;
012	0C	♀ (FF)	028	1C	∟ (FS)	044	2C	,	060	3C	<
013	0D	♪ (CR)	029	1D	↔ (GS)	045	2D	—	061	3D	=
014	0E	♫ (SO)	030	1E	▲ (RS)	046	2E	.	062	3E	>
015	0F	☼ (SI)	031	1F	▼ (US)	047	2F	/	063	3F	?

▶

Tab. A-1 Extended ASCII Character Set

DEC:	HEX	CHARACTER	DEC:	HEX	CHARACTER	DEC:	HEX	CHARACTER	DEC:	HEX	CHARACTER
064	40	@	080	50	P	096	60	‘	112	70	p
065	41	A	081	51	Q	097	61	a	113	71	q
066	42	B	082	52	R	098	62	b	114	72	r
067	43	C	083	53	S	099	63	c	115	73	s
068	44	D	084	54	T	100	64	d	116	74	t
069	45	E	085	55	U	101	65	e	117	75	u
070	46	F	086	56	V	102	66	f	118	76	v
071	47	G	087	57	W	103	67	g	119	77	w
072	48	H	088	58	X	104	68	h	120	78	x
073	49	I	089	59	Y	105	69	i	121	79	y
074	4A	J	090	5A	Z	106	6A	j	122	7A	z
075	4B	K	091	5B	[	107	6B	k	123	7B	{
076	4C	L	092	5C	\	108	6C	l	124	7C	¦
077	4D	M	093	5D	]	109	6D	m	125	7D	}
078	4E	N	094	5E	^	110	6E	n	126	7E	~
079	4F	O	095	5F	—	111	6F	o	127	7F	△ (DEL)

Tab. A-1 Extended ASCII Character Set (cont.)

DEC	HEX	CHARACTER	DEC	HEX	CHARACTER	DEC	HEX	CHARACTER	DEC	HEX	CHARACTER
128	80	Ç	144	90	É	160	A0	á	176	B0	░
129	81	ü	145	91	æ	161	A1	í	177	B1	▒
130	82	é	146	92	Æ	162	A2	ó	178	B2	▓
131	83	â	147	93	ô	163	A3	ú	179	B3	│
132	84	ä	148	94	Ö	164	A4	ñ	180	B4	┤
133	85	à	149	95	ò	165	A5	Ñ	181	B5	╡
134	86	å	150	96	û	166	A6	ª	182	B6	╢
135	87	ç	151	97	ù	167	A7	º	183	B7	╖
136	88	ê	152	98	ÿ	168	A8	¿	184	B8	╕
137	89	ë	153	99	Ö	169	A9	⌐	185	B9	╣
138	8A	è	154	9A	Ü	170	AA	¬	186	BA	║
139	8B	ï	155	9B	¢	171	AB	1/2	187	BB	╗
140	8C	î	156	9C	£	172	AC	1/4	188	BC	╝
141	8D	ì	157	9D	¥	173	AD	¡	189	BD	╜
142	8E	Ä	158	9E	Pt	174	AE	«	190	BE	╛
143	8F	Å	159	9F	f	175	AF	»	191	BF	┐

Tab. A-1 Extended ASCII Character Set (cont.)

DEC	HEX	CHARACTER	DEC	HEX	CHARACTER	DEC	HEX	CHARACTER	DEC	HEX	CHARACTER
192	C0	└	208	D0	╨	224	E0	α	240	F0	≡
193	C1	┴	209	D1	╤	225	E1	β	241	F1	±
194	C2	┬	210	D2	╥	226	E2	Γ	242	F2	≥
195	C3	├	211	D3	╙	227	E3	π	243	F3	≤
196	C4	─	212	D4	╘	228	E4	Σ	244	F4	⌠
197	C5	┼	213	D5	╒	229	E5	σ	245	F5	⌡
198	C6	╞	214	D6	╓	230	E6	µ	246	F6	÷
199	C7	╟	215	D7	╫	231	E7	τ	247	F7	≈
200	C8	╚	216	D8	╪	232	E8	Φ	248	F8	°
201	C9	╔	217	D9	┘	233	E9	Θ	249	F9	•
202	CA	╩	218	DA	┌	234	EA	Ω	250	FA	·
203	CB	╦	219	DB	█	235	EB	δ	251	FB	√
204	CC	╠	220	DC	▄	236	EC	∞	252	FC	n
205	CD	═	221	DD	▌	237	ED	∅	253	FD	²
206	CE	╬	222	DE	▐	238	EE	∈	254	FE	▮
207	CF	╧	223	DF	▀	239	EF	∩	255	FF	(SPACE) (SP)

Tab. A-1 Extended ASCII Character Set (cont.)

NATIONAL VARIATIONS IN EXTENDED ASCII CODE (ISO 8-bit code)

For Denmark, Norway, Greece and Portugal certain characters are displayed differently. These characters and their decimal and hexadecimal codes are shown in the following table.

DEC	HEX	CHARACTER	DEC	HEX	CHARACTER	DEC	HEX	CHARACTER	DEC	HEX	CHARACTER
128	80	Ç	144	90	É	160	A0	á	176	B0	░
129	81	ü	145	91	æ	161	A1	í	177	B1	▒
130	82	é	146	92	Æ	162	A2	ó	178	B2	▓
131	83	â	147	93	ô	163	A3	ú	179	B3	│
132	84	ä	148	94	Ö	164	A4	ñ	180	B4	┤
133	85	à	149	95	ò	165	A5	Ñ	181	B5	╡
134	86	å	150	96	û	166	A6	õ	182	B6	╢
135	87	ç	151	97	ù	167	A7	Õ	183	B7	╖
136	88	ê	152	98	ÿ	168	A8	¿	184	B8	╕
137	89	ë	153	99	Ö	169	A9	ã	185	B9	╣
138	8A	è	154	9A	Ü	170	AA	Ã	186	BA	║
139	8B	ï	155	9B	ø	171	AB	ℓ	187	BB	╗
140	8C	î	156	9C	£	172	AC	'n	188	BC	╝
141	8D	ì	157	9D	Ø	173	AD	¡	189	BD	╜
142	8E	Ä	158	9E	Ŀ	174	AE	^{3}	190	BE	╛
143	8F	Å	159	9F	I·	175	AF	¤	191	BF	┐

Tab. A-2 National Characters for Denmark and Norway

DEC	HEX	CHARACTER	DEC	HEX	CHARACTER	DEC	HEX	CHARACTER	DEC	HEX	CHARACTER
192	C0	└	208	D0	╨	224	E0	α	240	F0	≡
193	C1	┴	209	D1	╤	225	E1	β	241	F1	±
194	C2	┬	210	D2	╥	226	E2	Γ	242	F2	≥
195	C3	├	211	D3	╙	227	E3	π	243	F3	≤
196	C4	─	212	D4	╘	228	E4	Σ	244	F4	⌠
197	C5	┼	213	D5	╒	229	E5	σ	245	F5	⌡
198	C6	╞	214	D6	╓	230	E6	μ	246	F6	÷
199	C7	╟	215	D7	╫	231	E7	τ	247	F7	≈
200	C8	╚	216	D8	╪	232	E8	Φ	248	F8	°
201	C9	╔	217	D9	┘	233	E9	Θ	249	F9	•
202	CA	╩	218	DA	┌	234	EA	Ω	250	FA	·
203	CB	╦	219	DB	█	235	EB	δ	251	FB	√
204	CC	╠	220	DC	▄	236	EC	∞	252	FC	n
205	CD	═	221	DD	▌	237	ED	∅	253	FD	2
206	CE	╬	222	DE	▐	238	EE	∈	254	FE	▮
207	CF	╧	223	DF	▀	239	EF	∩	255	FF	(SPACE) (SP)

Tab. A-2 National Characters for Denmark and Norway (cont.)

DEC	HEX	CHARACTER	DEC	HEX	CHARACTER	DEC	HEX	CHARACTER	DEC	HEX	CHARACTER
128	80	Α	144	90	Ρ	160	A0	ι	176	B0	░
129	81	Β	145	91	Σ	161	A1	κ	177	B1	▒
130	82	Γ	146	92	Τ	162	A2	λ	178	B2	▓
131	83	Δ	147	93	Υ	163	A3	μ	179	B3	│
132	84	Ε	148	94	Φ	164	A4	ν	180	B4	┤
133	85	Ζ	149	95	Χ	165	A5	ξ	181	B5	╡
134	86	Η	150	96	Ψ	166	A6	ο	182	B6	╢
135	87	Θ	151	97	Ω	167	A7	π	183	B7	╖
136	88	Ι	152	98	α	168	A8	ρ	184	B8	╕
137	89	Κ	153	99	β	169	A9	σ	185	B9	╣
138	8A	Λ	154	9A	γ	170	AA	ς	186	BA	║
139	8B	Μ	155	9B	δ	171	AB	τ	187	BB	╗
140	8C	Ν	156	9C	ε	172	AC	υ	188	BC	╝
141	8D	Ξ	157	9D	ζ	173	AD	φ	189	BD	╜
142	8E	Ο	158	9E	η	174	AE	χ	190	BE	╛
143	8F	Π	159	9F	θ	175	AF	ψ	191	BF	┐

Tab. A-3 National Characters for Greece

DEC	HEX	CHARACTER	DEC	HEX	CHARACTER	DEC	HEX	CHARACTER	DEC	HEX	CHARACTER
192	C0	└	208	D0	╨	224	E0	ω	240	F0	Ω
193	C1	┴	209	D1	╤	225	E1	ά	241	F1	±
194	C2	┬	210	D2	╥	226	E2	έ	242	F2	≥
195	C3	├	211	D3	╙	227	E3	ή	243	F3	≤
196	C4	─	212	D4	╘	228	E4	ϊ	244	F4	⌠
197	C5	┼	213	D5	╒	229	E5	ί	245	F5	⌡
198	C6	╞	214	D6	╓	230	E6	ό	246	F6	÷
199	C7	╟	215	D7	╫	231	E7	ύ	247	F7	≈
200	C8	╚	216	D8	╪	232	E8	ϋ	248	F8	°
201	C9	╔	217	D9	┘	233	E9	ώ	249	F9	£
202	CA	╩	218	DA	┌	234	EA	Ά	250	FA	•
203	CB	╦	219	DB	█	235	EB	Έ	251	FB	√
204	CC	╠	220	DC	▄	236	EC	Ή	252	FC	n
205	CD	═	221	DD	▌	237	ED	Ί	253	FD	²
206	CE	╬	222	DE	▐	238	EE	Ό	254	FE	■
207	CF	╧	223	DF	▀	239	EF	Ύ	255	FF	(SPACE) (SP)

Tab. A-3 National Characters for Greece (cont.)

DEC	HEX	CHARACTER	DEC	HEX	CHARACTER	DEC	HEX	CHARACTER	DEC	HEX	CHARACTER
128	80	Ç	144	90	É	160	A0	á	176	B0	░
129	81	ü	145	91	æ	161	A1	í	177	B1	▒
130	82	é	146	92	Æ	162	A2	ó	178	B2	▓
131	83	â	147	93	ô	163	A3	ú	179	B3	│
132	84	ä	148	94	ö	164	A4	ñ	180	B4	┤
133	85	à	149	95	ò	165	A5	Ñ	181	B5	╡
134	86	å	150	96	û	166	A6	õ	182	B6	╢
135	87	ç	151	97	ù	167	A7	Õ	183	B7	╖
136	88	ê	152	98	ÿ	168	A8	¿	184	B8	╕
137	89	ë	153	99	Ö	169	A9	ã	185	B9	╣
138	8A	è	154	9A	Ü	170	AA	Ã	186	BA	║
139	8B	ï	155	9B	Á	171	AB	Ú	187	BB	╗
140	8C	î	156	9C	£	172	AC	í	188	BC	╝
141	8D	ì	157	9D	À	173	AD	¡	189	BD	╜
142	8E	Ä	158	9E	Ê	174	AE	³	190	BE	╛
143	8F	Å	159	9F	Ô	175	AF	Ó	191	BF	┐

Tab. A-4 National Characters for Portugal

DEC	HEX	CHARACTER	DEC	HEX	CHARACTER	DEC	HEX	CHARACTER	DEC	HEX	CHARACTER
192	C0	└	208	D0	╨	224	E0	∝	240	F0	≡
193	C1	┴	209	D1	╤	225	E1	β	241	F1	±
194	C2	┬	210	D2	╥	226	E2	Γ	242	F2	≥
195	C3	├	211	D3	╙	227	E3	π	243	F3	≤
196	C4	─	212	D4	╘	228	E4	Σ	244	F4	⌠
197	C5	┼	213	D5	╒	229	E5	σ	245	F5	⌡
198	C6	╞	214	D6	╓	230	E6	µ	246	F6	÷
199	C7	╟	215	D7	╫	231	E7	τ	247	F7	≈
200	C8	╚	216	D8	╪	232	E8	Φ	248	F8	°
201	C9	╔	217	D9	┘	233	E9	Θ	249	F9	●
202	CA	╩	218	DA	┌	234	EA	Ω	250	FA	•
203	CB	╦	219	DB	█	235	EB	δ	251	FB	√
204	CC	╠	220	DC	▄	236	EC	∞	252	FC	n
205	CD	═	221	DD	▌	237	ED	∅	253	FD	2
206	CE	╬	222	DE	▐	238	EE	∈	254	FE	■
207	CF	╧	223	DF	▀	239	EF	∩	255	FF	(SPACE) (SP)

Tab. A-4 National Characters for Portugal (cont.)

EXTENDED KEYBOARD CODES

Certain keys and key combinations do not produce ASCII code (one byte). Instead they produce two bytes; the first byte is always zero. This zero value indicates an extended keyboard code. The following table shows the value of the second byte, when these key(s) are pressed.

DECIMAL	HEXADECIMAL	CHARACTER
3	03	(Nul) NUL
15	0F	Cursor Left
16	10	ALT and Q
17	11	ALT and W
18	12	ALT and E
19	13	ALT and R
20	14	ALT and T
21	15	ALT and Y
22	16	ALT and U
23	17	ALT and I
24	18	ALT and O
25	19	ALT and P
30	1E	ALT and A
31	1F	ALT and S
32	20	ALT and D
33	21	ALT and F
34	22	ALT and G
35	23	ALT and H
36	24	ALT and J
37	25	ALT and K
38	26	ALT and L
44	2C	ALT and Z
45	2D	ALT and X
46	2E	ALT and C
47	2F	ALT and V
48	30	ALT and B
49	31	ALT and N
50	32	ALT and M
59	3B	F1
60	3C	F2
61	3D	F3
62	3E	F4
63	3F	F5
64	40	F6
65	41	F7
66	42	F8
67	43	F9
68	44	F10
71	47	HOME

DECIMAL	HEXADECIMAL	CHARACTER
72	48	Cursor up
73	49	PGUP
75	4B	Cursor Left
77	4D	Cursor Right
79	4F	END
80	50	Cursor Down
81	51	PGDN
82	52	INS
83	53	DEL
84	54	SHIFT and F1
85	55	SHIFT and F2
86	56	SHIFT and F3
87	57	SHIFT and F4
88	58	SHIFT and F5
89	59	SHIFT and F6
90	5A	SHIFT and F7
91	5B	SHIFT and F8
92	5C	SHIFT and F9
93	5D	SHIFT and F10
94	5E	CTRL and F1
95	5F	CTRL and F2
96	60	CTRL and F3
97	61	CTRL and F4
98	62	CTRL and F5
99	63	CTRL and F6
100	64	CTRL and F7
101	65	CTRL and F8
102	66	CTRL and F9
103	67	CTRL and F10
104	68	ALT and F1
105	69	ALT and F2
106	6A	ALT and F3
107	6B	ALT and F4
108	6C	ALT and F5
109	6D	ALT and F6
110	6E	ALT and F7
111	6F	ALT and F8
112	70	ALT and F9
113	71	ALT and F10

▶

DECIMAL	HEXADECIMAL	CHARACTER
114	72	CTRL and PRTSC
115	73	CTRL and Cursor Left
116	74	CTRL and Cursor Right
117	75	CTRL and END
118	76	CTRL and PGDN
119	77	CTRL and HOME
120	78	ALT and 1
121	79	ALT and 2
122	7A	ALT and 3
123	7B	ALT and 4
124	7C	ALT and 5
125	7D	ALT and 6
126	7E	ALT and 7
127	7F	ALT and 8
128	80	ALT and 9
129	81	ALT and 0
130	82	ALT and -
131	83	ALT and =
132	84	CTRL and PGUP

Note

Codes 120 to 131 decimal refer to the top row of keys in the body of the keyboard, not those on the right hand keypad.

The **SHIFT**, **CTRL** and **ALT** keys do not generate specific codes; they are used to modify the codes generated by other keys.

Note for Programmers

The above code is contained in the AH register. For extended keyboard codes 00 Hexadecimal is contained in the AL register. For displayable ASCII characters, normally the AH register contains the keyboard scan code and the AL register contains the ASCII code. See Appendix B of the MS-DOS System Programmer Guide for more details of National Keyboard Layouts and Codes.

B. ANSI ESCAPE SEQUENCES

ABOUT THIS APPENDIX

This appendix describes the ANSI escape sequences.

CONTENTS

INTRODUCTION B-1

CURSOR MOVING FUNCTIONS B-2

ERASING FUNCTIONS B-4

GRAPHIC MODE FUNCTIONS B-5

KEY REASSIGNMENT B-7

INTRODUCTION

An ANSI escape sequence is a series of characters beginning with the character **ESC** (decimal 27, hex 1B) that can be generated to define functions to MS-DOS. Specifically you can affect cursor movement, erase specific areas of the screen and set the graphics mode, by issuing these sequences through standard MS-DOS input/output.

The sequences described in this appendix require the presence of the ANSI console driver. ANSI.SYS is a file included on your MS-DOS system diskette. To install the ANSI console driver the following command must be placed in the CONFIG.SYS file:

DEVICE= [*d*:][*path*] ANSI.SYS

Refer to Appendix C for information about the CONFIG.SYS file and the DEVICE command.

Remarks

The following notes are general to all escape sequences:

1. ESC can be generated in a variety of ways: changing your prompt (PROMPT=$e); using the Video File Editor (see Chapter 7); or by program.

2. *Pn* represents a numeric parameter. This is a decimal number.

3. *Ps* represents a selective parameter. The parameter is still a decimal integer but is one that must be selected from a list of alternatives.

4. Where no parameter is specified, or where zero is entered, a default value is assumed.

5. No spaces should be typed inside escape sequences; any space shown in the specification is purely for ease of reading.

CURSOR MOVING FUNCTIONS

The following escape sequences affect the position of the cursor on the screen.

MNEMONIC	SEQUENCE	MEANING
CUP (Cursor Position) or HVP E (Horizontal and Vertical Position)	ESC [*Pn* ; *Pn* H or ESC [*Pn* ; *Pn* f	The cursor is moved to the line and column specified by the first and second parameters, respectively. The default values are 1. If no parameters are specified the cursor is moved to the home position.
CUU (Cursor Up)	ESC [*Pn* A	Moves the cursor up the screen by the number of rows specified by the parameter. If no parameter is specified one line is assumed. No action is taken if the cursor is already on the top line of the screen.
CUD (Cursor Down)	ESC [*Pn* B	Moves the cursor down the screen by the number of rows specified by the parameter. If no parameter is specified one line is assumed. No action is taken if the cursor is already on the bottom line of the screen.

MNEMONIC	SEQUENCE	MEANING
CUF (Cursor Forward)	ESC [*Pn* C	Moves the cursor to the right by the number of columns specified by the parameter. If no parameter is specified then one column is assumed. No action is taken if the cursor is already on the right-most column.
CUB (Cursor Backward)	ESC [*Pn* D	Moves the cursor left by the number of columns specified by the parameter. If no parameter is specified then one column is assumed. No action is taken if the cursor is already on the left-most column.
DSR (Device Status Report)	ESC [6 n	Causes the console driver to perform a CPR (Cursor Position Report) sequence.
CPR (Cursor Position Report)	ESC [*Pn* ; *Pn* R	The current cursor position is reported via the standard input/output device. The first parameter specifies the line and the second specifies the column. This sequence is performed by the console driver on receipt of a DSR sequence.

▶

MNEMONIC	SEQUENCE	MEANING
SCP (Save Cursor Position)	ESC [s	The current cursor position is saved. The saved value can subsequently be restored by issuing an RCP (Restore Cursor Position) sequence.
RCP (Restore Cursor Position)	ESC [u	The current cursor position is restored to what it was at the time that the console Driver received the SCP sequence.

ERASING FUNCTIONS

The following sequences erase specific areas of the video display.

MNEMONIC	SEQUENCE	MEANING
ED (Erase Display)	ESC [2 J	The screen is erased and the cursor is moved to the home position.
EL (Erase Line)	ESC [K	The current line from the cursor position to the end of the line is erased, including the cursor position itself.

GRAPHIC MODE FUNCTIONS

MNEMONIC	SEQUENCE	MEANING
SGR (Set Graphics Rendition)	ESC [*Ps* ; ... ;*Ps* m	The graphic functions specified by the parameters are invoked. The functions set by this sequence remain in effect until another SGR sequence is issued. The parameter values are as follows: 0 - all attributes off (normal display) 1 - high intensity (bold) 5 - sets blink on 7 - reverse video on 8 - concealed on (makes display invisible) 30 - black foreground 31 - red foreground 32 - green foreground 33 - yellow foreground 34 - blue foreground on color display; underline on monochrome display 35 - magenta foreground 36 - cyan foreground

▶

MNEMONIC	SEQUENCE	MEANING
		37 -white foreground 40 - black background 41 - red background 42 -green background 43 -yellow background 44 -blue background 45 -magenta background 46 -cyan background 47 -white background
SM (Set Mode)	ESC [= *Ps* h	The screen width and type are set by the parameter specified. The possible parameter values are as follows: 0 - 40 column by 25 line black and white 1 - 40 column by 25 line color 2 - 80 column by 25 line black and white 3 - 80 column by 25 line color 4 - 320 x 200 color

MNEMONIC	SEQUENCE	MEANING
		5 - 320 x 200 black and white 6 - 640 x 200 black and white 7 - automatic text wrap at end-of-line
RM (Reset Mode)	ESC [= *Ps*l	Resets the attribute set by the SM sequence. Parameter values are the same as for the SM sequence.
Except for		7 - no text wrap at end-of-line (excess characters are not displayed)

KEY REASSIGNMENT

An ANSI escape sequence can be used to assign an ASCII code, a string, or a combination of ASCII codes and strings to any key or value or valid key-stroke combination.

Valid control sequences are:

ESC [*Pn* ; *Pn* ;... p

ESC [*Pn* ; ''*string*'' p

ESC [*Pn* ; ''*string*''; *Pn*; *Pn*; ''*string*''; *Pn* p

or any other combination of strings and decimal numbers.

The first parameter defines which ASCII code is to be mapped, unless it is zero, in which case the first and second parameters comprise the extended keyboard code. Refer to Appendix A for a complete list of ASCII and extended keyboard codes.

Note that the ASCII codes must be entered in decimal.

Examples

If your program issues the sequence...	**THEN...**
ESC[66;82p	pressing **B** becomes R instead of B.
ESC[0;67;"dir";13p	a string "dir" is entered followed by a carriage return whenever you press **F9**. The initial 0 indicates that the **F9** function key is represented by an extended keyboard code 67.

C. CONFIGURING MS-DOS

ABOUT THIS APPENDIX

This appendix describes how you can configure MS-DOS to suit the requirements of your application.

CONTENTS

INTRODUCTION	C-1
CONFIGURING COMMANDS	C-1
BREAK	C-1
BUFFERS	C-2
COUNTRY	C-3
DEVICE	C-5
FCBS	C-5
FILES	C-6
LASTDRIVE	C-7
SHELL	C-8

INTRODUCTION

Whenever MS-DOS is initialized, it searches the root directory for a file named CONFIG.SYS. If this file exists, it is read and the configuring commands within it are executed. If CONFIG.SYS is not found the default setting for each command is used.

You can create or modify the CONFIG.SYS file using the Video File Editor or Line Editor, and include in it any of the commands described in this appendix.

Also if the particular command within CONFIG.SYS is not found its default setting is used.

CONFIGURING COMMANDS

BREAK

Sets or resets the **CTRL C** break facility.

BREAK = ON | OFF

Characteristics

The action of the BREAK command is exactly the same as described in Chapter 6. However, it can only occur once in the CONFIG.SYS file. The state set by including this command in the CONFIG.SYS file can later be reversed by entering a BREAK command with the opposite argument at the keyboard.

The default is OFF.

BUFFERS

Sets the number of buffers to be allocated.

BUFFERS= *number*

Where

SYNTAX ELEMENT	MEANING
number	A decimal number in the range 1 to 99 that defines the number of buffers to be allocated. The default is 5.

Characteristics

The optimum number of buffers depends on factors such as:

- the type of applications most commonly used
- the amount of memory

If your applications perform a lot of random accesses you will probably benefit by using a larger number of buffers. However, beyond a certain number it may take MS-DOS longer to scan the buffers than it would to access a value from disk. For most data base applications between 10 and 20 buffers is a reasonable number.

Conversely, applications that perform mostly sequential accesses gain little by increasing the number of buffers beyond the default of five.

Memory size affects the optimum number of buffers because each additional buffer requires 528 bytes of memory.

The optimum number of buffers can only be determined by trial and error.

COUNTRY

Sets the country to allow MS-DOS to use the correct national time, date, currency and decimal separators.

COUNTRY= *number*

Where

SYNTAX ELEMENT	MEANING
number	A three digit number which is the telephonic international country code: 001 United States of America 031 Netherlands 032 Belgium 033 France 034 Spain 039 Italy 041 Switzerland 044 United Kingdom 045 Denmark 047 Norway 049 Germany 061 Australia 358 Finland 972 Israel

Note: If your country is not supported, choose the country which uses your national conventions.

The default is COUNTRY = 001

DEVICE

Installs a device driver.

DEVICE = *pathname*

Where

SYNTAX ELEMENT	MEANING
pathname	The file containing the device driver with an optional drive and path specifying the containing directory.

Characteristics

If *pathname* is the file ANSI.SYS, the escape sequences described in Appendix B are supported.

If *pathname* is VDISK.SYS, refer to Appendix F for further details.

Alternatively, you may enter the file name of any device driver written for your system.

FCBS

Defines the number of FILES opened with File Control Blocks that can be open at any one time.

FCBS = *maxopen,number*

Where

SYNTAX ELEMENT	MEANING
maxopen	This is a number from 1 to 255 which represents the maximum number of files that can be opened with File Control Blocks. The default value is 4.
number	This is a number from 0 to 255 that specifies the files that MS-DOS cannot close automatically if the application tries to open more than *maxopen*. the first files opened are the protected files. The default value of *number* is 0.

Note: You are advised not to use this command unless the application specifies or you are having trouble while file-sharing. If sharing is not loaded this command is not applicable.

FILES

Defines the number of file handles that can be open at any one time.

FILES = *number*

Where

SYNTAX ELEMENT	MEANING
number	A decimal number in the range 1 to 255 defining the number of files that can be open concurrently. The default is 8.

Characteristics

Each additional file above the default value of 8 requires an extra 39 bytes of memory.

You should increase the value only if you have an application that gives an error message indicating a shortage of open file handles.

LASTDRIVE

Sets the maximum number of drives that you may access

LASTDRIVE = *drive-letter*

Where

SYNTAX ELEMENT	MEANING
drive-letter	Any letter from A through Z, defining the last valid drive that MS-DOS will accept.

Characteristics

The default is:

```
LASTDRIVE = E
```

In a network environment, you may use the REDIR command to assign the extra drive letters to a volume on a remote computer.

Remarks

You cannot set LASTDRIVE less than the number of drives on your computer.

SHELL

Loads a top-level command processor, which can be COMMAND.COM or an alternative command processor.

SHELL = *pathname path* **/P**

Where

SYNTAX ELEMENT	MEANING
pathname	The drive, path and file name containing the top-level command processor to be loaded.
path	The drive and path of the directory containing the top-level command processor. This sets the environment COMSPEC= to point to the command process for reloading purposes.
/P	This switch must be specified. This is to indicate that the command processor is to be used for system startup. If omitted AUTOEXEC.BAT will not be executed and an EXIT causes a system crash.

For example:

IF you enter...	THEN...
SHELL=C:\COMMAND.COM C:\BIN /P	this command causes the top level command processor to be loaded from the root directory of the C: drive. The second parameter indicates that the transitory part of the command processor is to be reloaded from the BIN directory of the C: drive. The **/P** switch must be specified to indicate this is the top level command processor.

Remarks

If you are writing an alternative command processor remember to duplicate COMMAND.COM's internal commands, batch processor and program loader.

D. ERROR MESSAGES

ABOUT THIS APPENDIX

This appendix explains the various error messages that can be displayed by MS-DOS and its utilities.

CONTENTS

ERROR MESSAGES **D-1**

This appendix lists in alphabetical order the error messages you may encounter when using MS-DOS.

Device Errors

Errors may occur when reading from or writing to devices such as disks or printers. These errors cause the system to stop and output a message of this form.

> *type* error reading/writing *device*
> Abort, Retry, Ignore?

where *type* represents one of the following:

> Bad call format error
> Bad command error
> Bad unit error
> Data error
> FCB unavailable
> General failure
> Invalid disk change
> Lock violation
> No paper error
> Non-DOS disk error
> Not ready error
> Read fault error
> Sector not found error
> Seek error
> Sharing violation
> Write fault error
> Write protect error

device indicates the driver or disk device in error. See the section RESERVED DEVICE NAMES in chapter 4 for a list of device names. Disk drives are indicated by a single letter.

When you receive one of these messages, do one of the following:

> Enter **A** for Abort. The system ends the program that requested the read or write.

Enter **R** for Retry. The system tries the read or write operation again.

Enter **I** for Ignore. The system ignores the error and attempts to continue the program. (This method may cause loss of data).

These device error messages are described along with the rest of the error messages in the following table:

ERROR MESSAGE	MEANING
Abort edit (Y/N)? (EDLIN)	MS-DOS displays this message when you choose the Q (Quit) command in EDLIN. The Quit command exits the editing session without saving any editing changes. Specify **Y** (for "Yes") or **N** (for "No").
Access denied (MS-DOS)	MS-DOS displays this message when you tried to write to or delete a file marked as read only. If you really want to carry out this action, use the ATTRIB command to give the file a read/ write attribute.
All files canceled by operator (PRINT)	MS-DOS displays this message when you specify the **/T** switch with the PRINT command.
All partitions are currently in use (FDISK)	Self-explanatory.
All specified files are contiguous (CHKDSK)	All files are allocated contiguously on the disk without fragmentation.

ERROR MESSAGE	MEANING
Allocation error, size adjusted (CHKDSK)	The size of the file indicated in the directory was not consistent with the amount of data actually allocated to the file. Adjustment only actually takes place, if you specify the **/F** switch with CHKDSK; the file is truncated at the end of the last valid cluster.
Ambiguous switch: z (LINK)	The characters in *z* identify more than one linker parameter.
Amount read less than size in header (EXE2BIN)	The file is smaller than its header indicates. Recompile (or reassemble) and relink the program.
An internal failure has occurred (LINK)	The linker program has failed. Report the conditions of the failure to your Olivetti dealer.
Attempt to access data outside of segment bounds, possibly bad object module (LINK).	An invalid object file has been specified.
Attempted write-protect violation (FORMAT)	You cannot format a write-protected diskette. Use another disk or remove the write-protection tag.
Backup file sequence error (RESTORE)	The file being restored is backed up on more than one diskette. The wrong diskette in the sequence has been inserted.

ERROR MESSAGE	MEANING
Bad call command error (device error).	A request header of incorrect length was passed to a device driver. Contact your Olivetti dealer.
Bad command or file name (MS-DOS)	The command cannot find the program you asked it to run. You either mistyped the filename or the file does not exist in the specified disk directories.
Bad file (FC)	One of the files you specified is defective.
Bad numeric parameter (LINK)	The value specified with the /STACK parameter is not a valid numeric constant.
Bad or missing Command Interpreter (MS-DOS)	The MS-DOS disk being loaded does not contain the file COMMAND.COM in the root directory. This message may also appear if an error occurs during loading of the system disk or if the COMSPEC = *parameter* does not point to a directory containing COMMAND.COM.
Bad or missing *filename* (MS-DOS)	One of the following conditions occurred during startup: The device driver named in the DEVICE = *parameter* does not exist in CONFIG.SYS.

ERROR MESSAGE	MEANING
Bad or missing *filename* (MS-DOS) (cont.)	A break address has been set which is out of bounds for the machine. An error occurred during loading of the driver.
Bad unit error (device error)	An invalid subunit number has been sent to a device driver. Contact your dealer.
Batch file missing (MS-DOS)	A file being processed in batch mode is no longer present. It may have been removed or erased during processing. The batch stops and returns control to MS-DOS.
BF (DEBUG)	The specified flag code setting is invalid. Renter the Register (RF) command with the correct code.
BP (DEBUG)	More than ten breakpoints were specified for the Go (G) command. Reenter the GO command with ten or fewer breakpoints.
BR (DEBUG)	An invalid register name has been specified. Reenter the Register (R) command with a valid register name.
Cannot CHDIR to root. Processing cannot continue (CHKDSK)	The disk you are checking is faulty. Reboot MS-DOS and try to RECOVER the disk.

▶

ERROR MESSAGE	MEANING
Cannot CHKDSK a Network drive (CHKDSK)	You cannot check drives which are redirected over the Network.
Cannot CHKDSK a SUBSTed or ASSIGNed drive (CHKDSK)	You cannot check drives which are SUBSTed or ASSIGNed.
Cannot do binary reads from a device (COPY)	You have tried to use the **/B** switch with the name of a device. Place a **/A** switch after the device name to copy in ASCII mode.
Cannot edit .BAK file-rename file (EDLIN)	You attempted to edit a backup copy created by EDLIN. Either rename the file or copy the .BAK file and give it a different extension.
Cannot find file object file. Change diskette hit ENTER (LINK)	The specified object module is not present on the diskette.
Cannot find library file. ENTER new drive letter: (LINK)	The specified library is not present on the current drive. Enter the drive containing the library.
Cannot format an ASSIGNED or SUBSTed drive (FORMAT)	You attempted to format a drive which is actually mapped to another drive by the ASSIGN or SUBST command. Run ASSIGN or SUBST again and clear all drive mappings.
Cannot FORMAT a Network drive (FORMAT)	You cannot format drives that are redirected over the Network.

ERROR MESSAGE	MEANING
Cannot load COMMAND, system halted (MS-DOS)	One of the following conditions occurred while loading the command processor: • The available memory map has been destroyed. • The command processor specified by the COMSPEC parameter does not exist. Reboot MS-DOS.
Cannot nest response file (LINK)	It is not possible to use a *@filespec* within an automatic response file.
Cannot open filename (PRINT)	Either MS-DOS cannot find the specified file to print or the file does not exist. Check the command for a valid filename.
Cannot open list file (LINK)	A list file cannot be opened because the disk or directory is full.
Cannot open overlay (LINK)	An overlay cannot be opened because the disk or directory is full.
Cannot open response file (LINK)	The specified response file does not exist.

ERROR MESSAGE	MEANING
Cannot open temporary file (LINK)	The directory or disk is full, hence the linker cannot create the VM.TMP file. Insert a new disk. Do not remove the disk that will receive the List.MAP file.
Cannot recover . entry processing continued (CHKDSK)	The . entry (working directory) is defective.
Cannot recover .. entry processing continued (CHDSK)	The .. entry (parent directory) is defective.
Cannot recover a Network drive (RECOVER)	You cannot recover files on drives that are redirected over the Network.
CHDIR .. failed, trying alternate method (CHKDSK)	In traveling the tree structure, CHKDSK was not able to return to a parent directory. It will try to return to that directory by starting over at the root and traveling down.
Compare error(s) (DISKCOMP)	Different information has been found on one or more disk locations.
Compare error at offset XXXXXXXX (COMP)	While comparing two files, different values were found at offset XXXXXXXX (hexadecimal). The values found are also displayed (in hexadecimal).
Compare more files (Y/N)? (COMP)	Answer **Y** if you wish to compare more files, otherwise enter **N**.

ERROR MESSAGE	MEANING
Contains XXX noncontiguous blocks (CHKDSK)	This message indicates that your files are fragmented. Fragmented files take longer to read COPY badly fragmented files to a newly formatted disk. Using the new disk will result in faster reading of the files.
Content of destination lost before copy (MS-DOS)	A file to be used as a source file to the Copy command has been overwritten prior to completion of the copy. Example: COPY F1+F2 F2 destroys F2 before it can be copied.
Data error (device error)	Data could not be read/written correctly because of a faulty disk.
Data left in (*filename*) (FC)	After reaching the end of one of the files in a file comparison, the other file still has uncompared data left.
DF (DEBUG)	Conflicting codes have been specified for a single flag. A flag can be changed only once for each Register (RF) command.
Directory entries adjusted (VDISK)	VDISK has adjusted the number of directory entries in the parameters of DEVICE=VDISK.SYS in the CONFIG.SYS command.
Directory error in TREE (TREE)	Self-explanatory.

ERROR MESSAGE	MEANING
Directory is joined, tree past this point not processed. (CHKDSK)	CHKDSK will not process directories which are joined.
Directory is totally empty, no . or .., tree past this point not processed. (CHKDSK)	The specified directory does not contain references to working and parent directories. Delete the specified directory and recreate it.
Directory not empty (JOIN)	You can only JOIN onto a directory which is empty.
Disk already has an MS-DOS partition (FDISK)	You cannot use the ''Create MS-DOS Partition'' option on a fixed disk which already has a DOS partition.
Disk boot failure (MS-DOS)	MS-DOS cannot be loaded from the disk in drive A. Retry, or if that fails, use another disk.
Disk error (device error)	An error has occurred reading from or writing to a disk.
Disk error reading FAT (CHKDSK)	An error occurred while CHKDSK was trying to update the file allocation table.
Disk error writing FAT (CHKDSK)	An error occurred while CHKDSK was trying to update the file allocation table.
Disk full. Edits lost (EDLIN)	EDLIN was not able to save your file due to lack of disk space.

ERROR MESSAGE	MEANING
Disk full-write not completed (EDLIN)	The disk you are using does not have enough space to save all the file. Part of the file may have been saved on disk, but the portion that has not been saved is lost.
Disk unsuitable for system disk (FORMAT)	The diskette contains a defective track where DOS files must reside. The disk may only be used for data.
Diskette is not a backup diskette (BACKUP & RESTORE)	The diskette in the drive was not created by BACKUP. The first file is not called BACKUP.@@@.
Diskette is not last backup diskette (BACKUP)	The diskette in the drive is not the last one in a series created by BACKUP.
Divide overflow (MS-DOS)	A divide by zero was attempted, or an internal logic error has occurred. The system continues as if **CTRL BREAK** had occurred.
Drive types or diskette types not compatible (DISKCOPY, DISKCOMP)	The source and target diskettes must have the same format capacity. See the respective command specifications.
(.)(..) Does not exist (CHKDSK)	This is an informational message from CHKDSK. This message indicates either the . or .. directory entry is invalid.

ERROR MESSAGE	MEANING
Duplicate filename or file not found (RENAME)	Either an attempt has been made to rename a file with a file name that already exists in the directory, or the file to be renamed could not be found on the specified (or default) drive.
Dup record too complex (LINK)	There is a problem with an object module created from an assembler source program. A single DUP requires 1024 bytes before expansion. Debug the source program then return to the linker.
End of input file (EDLIN)	The entire file was read into memory. If the file is read in sections, this message indicates the last section of the file is in memory.
Entry error (EDLIN)	You have not entered an EDLIN command correctly. Re-enter the command.
Entry has a bad allocation error, size adjusted (CHKDSK)	The file allocation table contains an invalid sector number. The file is truncated at the end of the last valid sector.
Entry has a bad attribute (or link or size) (CHKDSK)	This message may be preceded by one or two periods which indicate which subdirectory is invalid. If you have specified the **/F** switch, CHKDSK will try to correct the error.

ERROR MESSAGE	MEANING
Eof mark not found (COMP)	The end of valid data in the last block of two files being compared has not been found. Most likely to occur in non-text files.
Error in EXE file (MS-DOS)	The file contains erroneous relocation information created by LINK. The file may have been altered after creation.
Error in EXE/HEX file (DEBUG)	The EXE or HEX file contained invalid characters or records.
Error loading system from fixed disk (FDISK)	The operating system cannot be loaded from the fixed disk. Retry, or if that fails, boot the system from diskette and put a new copy of MSDOS onto the fixed disk using the SYS command.
Error reading drive *x* (RECOVER)	Self-explanatory.
Error reading file (PRINT)	Self-explanatory.
Error reading fixed disk (FDISK)	Five unsuccessful attempts have been made to read the startup record from fixed disk. Retry FDISK, of if that fails, consult your Installation and Operations Guide.
Error writing to device (COMMANDS)	The device issued an I/O error and your data was not written. Retry.

ERROR MESSAGE	MEANING
Error writing fixed disk (FDISK)	Five unsuccessful attempts, have been made to write the startup record on the fixed disk. Retry FDISK, or if that fails, consult your Installation and Operations Guide.
Errors found, F parameter not specified. Corrections will not be written to disk (CHKDSK)	As the **/F** parameter was not used, an analysis of the disk will be made and the results displayed, but no corrections will be written to the disk.
Errors on list device indicates that it may be offline. Please check it. (PRINT)	Your printer is off-line.
EXEC failure (MS-DOS)	One of the following conditions occurred while reading a file from disk: • Read error occurred. • The FILES= parameter in the configuration file is not large enough. Increase the value and restart MS-DOS.
EXE/HEX file cannot be written (DEBUG)	The data would require a backwards conversion that DEBUG does not support.
File allocation table bad for drive *x* Abort, Retry, Ignore? (MS-DOS, CHKDSK)	See DEVICE ERRORS at the beginning of this appendix. If the error persists, the diskette should be reformatted.

ERROR MESSAGE	MEANING
Files are different sizes (COMP)	The specified files are not of the same length and cannot be compared.
File *xxx* canceled by operator (PRINT)	When the operator cancels the printing, this message appears on the printer.
File cannot be converted (EXE2BIN)	The input file you have specified does not have the correct format for conversion.
File cannot be copied into itself (COPY)	A request was made to COPY a file and place the copy (with the same name) in the same directory as the source file. Either change the name given to the copy or put it on another diskette or directory.
File creation error (MS-DOS and commands)	An unsuccessful attempt was made to add a new file to the directory. Run CHKDSK to determine the cause of the error.
File is READ-ONLY (EDLIN)	You may not change this file because the file is designated read-only. If you really want to write to this file, use the ATTRIB command to give the file a read/write attribute.
File name must be specified (EDLIN)	You did not specify a file name when you started EDLIN.

ERROR MESSAGE	MEANING
File not found (COMMANDS)	The file you named in the Transfer command does not exist.
File not found (MS-DOS and commands)	A file named in a command does not exist on the disk in the specified (or default) drive.
File not in PRINT queue (PRINT)	The file you want to remove from the print queue is not in the queue.
File table bad (MS-DOS)	Self-explanatory. If this error is issued, you have to reboot the system.
FIND: File not found *filename* (FIND)	A non-existent file name was specified when issuing a FIND command.
FIND: Invalid number of parameters (FIND)	A string was not specified when issuing a FIND command.
FIND: Invalid parameter option-name (FIND)	You specified an invalid parameter to the FIND command.
FIND: Read error in filename (FIND)	An error occurred when FIND tried to read the file specified in the command.
FIND: Syntax error (FIND)	You entered an illegal string when issuing the FIND command.

ERROR MESSAGE	MEANING
First cluster number is invalid, entry truncated (CHKDSK)	An invalid pointer to the data area has been found in the file whose name proceeds this message. If **/F** was specified, the file is truncated to zero length.
Fixup offset exceeds field width (LINK)	An assembly language instruction refers to an address with a short instruction instead of a long instruction. Edit the assembler source program and process it again.
Fixups needed-base segment (hex): (EXE2BIN)	The source (.EXE) file contained information indicating that a load segment is required for the file. Specify the absolute segment address at which the finished module is to be located.
FOR cannot be nested (Batch)	It is not possible to have more than one FOR subcommand on one command line.
Format aborted (FORMAT)	Self-explanatory.
Format failure (FORMAT)	A disk error occurred on the diskette being formatted. The diskette cannot be formatted
Formatting target while copying (DISKCOPY)	Unformatted tracks have been found on the target diskette. These tracks will be formatted as copying proceeds.

▶

ERROR MESSAGE	MEANING
Has invalid cluster, file truncated (CHKDSK)	The specified file contains an invalid data area printer. If the **/F** parameter was used, the file is truncated at the last valid data block.
Illegal device name (MODE)	Permitted device names are: Printers LPT1, LPT2, LPT3. The Asynchronous Communications Adapter must be one of COM1 or COM2. Only one space is allowed between MODE and its parameter(s).
Illegal parameter (MODE)	Self-explanatory.
Incompatible system size (SYS)	The hidden files IO.SYS and MSDOS.SYS do not take up the same amount of space on the target diskette as the new system will need.
Incorrect DOS version (Various EXTERNAL MS-DOS COMMANDS)	Many DOS utilities will not run on older versions of MS-DOS. For example the utilities CHKDSK, PRINT, and SYS will only run under the exact version of MS-DOS with which they were distributed.
Incorrect number of parameters (JOIN) (SUBST)	You specified too many or too few options in the command line.

ERROR MESSAGE	MEANING
Incorrect parameter (ASSIGN SHARE)	One of the options you specified is wrong.
Input file read error (LINK)	An invalid object file has been entered in the command line.
Insufficient disk space (MS-DOS and commands)	The diskette does not contain enough free space to contain the new file.
Insufficient memory (COMMANDS)	There is not enough memory to perform the specified operation.
Insufficient memory for system transfer (FORMAT)	Your memory configuration is insufficient to transfer the MS-DOS system files IO.SYS and MSDOS.SYS (the **/S** switch).
Insufficient room in root directory Erase files in root and repeat CHKDSK (CHKDSK)	CHKDSK cannot create an entry in the root directory for saving lost chains as files (see message ''*X* lost clusters found in *Y* chains -. Convert lost chains to files (Y/N?)'' because the root directory is full. You should copy some files from the root directory to another disk, then reexecute CHKDSK.
Insufficient space on disk (DEBUG)	A write command was issued to a disk that does not have enough free space to hold the data being written.

▶

ERROR MESSAGE	MEANING
Internal error (FC)	This message indicates an internal logic error in the FC utility. If this error is returned, you have to reboot the system.
Intermediate file error during pipe (MS-DOS)	One of the following errors has occurred: • One or both intermediate MS-DOS files cannot be created because the root directory is full. • The piping files do not exist. • There is insufficient space on the disk for the data being piped. Remove some files from the root directory and retry. If this fails, a piping file has been erased. Correct the program and retry.
Invalid characters in volume label (FORMAT)	Volume labels may contain up to 11 printable characters without a period (.).
Invalid COMMAND.COM in drive *X* (MS-DOS)	The program you have just run used up almost all of memory. MS-DOS must now reload the transient part of COMMAND.COM file from disk. However, MS-DOS cannot find COMMAND.COM on the disk or the copy found is invalid. Insert a disk into the X: drive which contains the same version of COMMAND.COM as with which you started MS-DOS. Press any key to commence the reloading.

▶

ERROR MESSAGE	MEANING
Invalid country code (MS-DOS) (SELECT)	You have specified a country number in your SELECT command or CONFIG.SYS file which is not configured in this implementation of MS-DOS. Country codes must be in the range 1-99 and are the same as the International dialling code for the selected country.
Invalid current directory. Processing cannot continue (CHKDSK)	CHKDSK has found an error in the disk's current directory. Restart the system and rerun CHKDSK.
Invalid date (MS-DOS) (DATE)	You specified an invalid date in response to the date prompt when starting MS-DOS or when using the DATE command.
Invalid device (CTTY)	The specified device name is invalid in MS-DOS
Invalid directory (MS-DOS)	The directory you specified either does not exist or is invalid. Check to see that you entered the directory name correctly.
Invalid drive in search path (MS-DOS)	One of the paths specified in the PATH command contains an invalid drive name. This error occurs during execution, not during the PATH command.
Invalid drive or filename (EDLIN)	You did not specify a valid drive or file name when involivng EDLIN.

ERROR MESSAGE	MEANING
Invalid drive specification (MS-DOS and commands)	You have tried to enter an invalid drive specifier in a command line.
Invalid drive was specified (MS-DOS and commands)	Self-explanatory.
Invalid environment (MS-DOS)	Self-explanatory.
Invalid format file (LINK)	An error has been found in a library.
Invalid handle (MS-DOS)	Self-explanatory.
Invalid memory block address (MS-DOS)	Self-explanatory.
Invalid number of parameters (Commands)	You have specified the wrong number of parameters on the command line. Check the syntax of the command you are using.
Invalid numeric parameters (LINK)	A character other than a digit has been included in a numeric parameter.
Invalid numeric switch specification switch error: ''*s:XXX*'' (LINK)	You specified a character for a switch requiring a numeric parameter. LINK will abort.
Invalid object module (LINK)	Object module(s) are incorrectly formed or incomplete.

ERROR MESSAGE	MEANING
Invalid parameter (Commands)	The parameter entered for a command was incorrect. Check the syntax of the command you are using.
Invalid parameters (MODE)	This message indicates one of: • No parameters entered. • First letter not L or C. • First parameter not one of: 40, 80, BW40, BW80, CO40, CO80, MONO, L, R. • The referenced display adapter is not present.
Invalid partition table on fixed disk (FDISK)	Invalid partition information has been detected during startup from fixed disk. Restart DOS using diskette and correct the fixed disk partition information using the FDISK command.
Invalid path (TREE)	A directory whose name is in another directory cannot be accessed by TREE. Use CHKDSK to determine the error in the directory structure.
Invalid path, not directory, or directory not empty (RMDIR)	The directory cannot be removed as the path contains an invalid name, or the directory is not empty. The current directory cannot be removed.

ERROR MESSAGE	MEANING
Invalid path, or filename (COPY)	A directory or filename that does not exist has been specified.
Invalid subdirectory entry (CHKDSK)	The subdirectory whose name precedes this message contains invalid information. For more detailed information rerun CHKDSK with the **/V** switch.
Invalid switch: *z* (LINK)	The characters in *z* do not identify a valid linker parameter.
Invalid switch specification (FORMAT)	Self-explanatory.
Invalid time (MS-DOS) (TIME)	You specified an invalid time in response to the time prompt when starting MS-DOS or using the TIME command.
Invalid working directory (MS-DOS)	Your disk is bad. Replace the disk or make another copy from your backup system disk.
filename is cross linked on cluster (CHKDSK)	You have two files cross linked. Make a copy of the file you want to keep, and then delete both files that are cross linked.
Label not found (Batch)	A non-existent label has been specified in a GOTO command within the batch file.

▶

ERROR MESSAGE	MEANING
Line too long (EDLIN)	During a Replace (R) command, the string given as the replacement caused the line to expand beyond the 253-character limit. The Replace command is ended abnormally. Split the long line into shorter lines; then reissue the Replace command.
List too long (EDLIN)	The list device you have specified is invalid. Subsequent attempts will return the same message until a valid device is specified.
Load1, system halted	This message is issued in extreme circumstances, when the system actually halts itself. If this error is returned, you have to reboot the system.
Memory allocation error. Cannot load COMMAND, system halted (MS-DOS)	The available memory map has been destroyed. You must restart MS-DOS.
Memory control blocks destroyed (MS-DOS)	Self-explanatory.
MS-DOS incorrect on default drive (Startup)	Self-explanatory.
MS-DOS not found on default drive (Startup)	Self-explanatory.

ERROR MESSAGE	MEANING
Must specify destination number (EDLIN)	A destination line number was not specified for a Copy or Move command. Reenter the command correctly.
New file (EDLIN)	This message is printed if EDLIN does not find a file with the name you specified if you are creating a new file, ignore this message. If you do not intend to create a new file, check to see that you correctly typed the filename of the file you wish to edit.
No fixed disks present (FDISK)	You cannot run the FDISK program due to one of the following condition: • No fixed disk attached. • Fixed disk is present in the expansion unit but the unit is not switched on. • Fixed disk is incorrectly installed.
No free file handles Cannot start COMMAND. COM, exiting (MS-DOS)	Reload MS-DOS. If this message persists, increase the size of the FILES= *parameter* in the CONFIG.SYS file, and reload MS-DOS.
No free handles (MS-DOS)	You have tried to load a second copy of the command processor, but too many files are currently open. Increase the size of the FILES= *parameter* in the configuration file and reload MS-DOS.

ERROR MESSAGE	MEANING
No MS-DOS partition to delete (FDISK)	You have used the Delete DOS Partition option when no such partition exists on the current fixed disk or you have tried to delete a non-existent partition
No MS-DOS partition Use FDISK to correct (FORMAT)	An attempt has been made to format a hard disk that does not have an MS-DOS partition.
Non-DOS disk (device error)	The file allocation table contains invalid information. The disk must be reformatted.
No object modules specified (LINK)	You have not specified any object modules for the linker
No operating system on fixed disk (FDISK)	Self-explanatory.
No paper (device error)	The printer is either not switched on or is out of paper.
No path (MS-DOS)	You typed PATH to display your search path. There is no current command search path.
No room in directory for file (EDLIN)	The directory of the specified diskette is already full, or the specified disk drive or file name is illegal.

ERROR MESSAGE	MEANING
No room in disk directory (DEBUG)	The directory of the specified diskette is already full.
No space for a *XXX* cylinder partition at cylinder *YYYY* (FDISK)	There is not enough space on disk to accommodate a partition with the specified number of cylinders at the specified position.
Non-System disk or disk error. Replace and strike any key when ready (Startup)	Restart the system with another MS-DOS disk in drive A.
Not enough memory (JOIN SHARE) (Other External Commands)	There is not enough memory for MS-DOS to run the command.
Not enough room for MS-DOS on this disk (SYS)	There is not enough room on the target disk for SYS to transfer the system files.
Not enough room to merge the entire file (EDLIN)	There was not enough room in memory to enable a Transfer command to merge the entire contents of a file.
Not found (EDLIN)	The search string was not found in the specified line range, or no further occurences were found after resuming.

▶

ERROR MESSAGE	MEANING
Not ready (PRINT)	If this message occurs when PRINT attempts a disk access, PRINT will keep trying until the drive is ready. Any other error causes the current file to be cancelled. An error message would be output on your printer in such a case.
Not ready error (device error)	The named device is not available for the read/ write operation required.
Out of environment space (MS-DOS)	There is not enough room in the program environment to accept more data.
Out of memory (MS-DOS and Commands)	Self-explanatory.
Out of space on list file (LINK)	There is not enough space on disk to hold the list file.
Out of space on run file (LINK)	There is not enough space on disk to hold the run file.
Out of space on VM.TMP (LINK)	There is not enough space on disk to allow the VM.TMP file to be expanded.
Path not found (PRINT)(MS-DOS)	You specified an invalid pathname.
Pathname too long (PRINT)(TREE)	The maximum pathname permitted is 63 characters.

ERROR MESSAGE	MEANING
PRINT queue is empty (PRINT)	There are no files waiting to be printed.
PRINT queue is full (PRINT)	You have tried to place more than ten files in the print queue.
Printer fault (MODE)	The printer mode has not been set because of one of the following conditions: • I/O error • Out of paper • Power off • Time out
Probable non-DOS disk. Continue (Y/N)? (CHKDSK)	The disk you are using is not recognized by this version of MS-DOS. The disk either was created by another system with a format that is not supported on this version of MS-DOS or is not an MS-DOS disk. Do not continue processing if CHKDSK returned this message for a removable disk. If this message is returned for a hard disk, the information describing the characteristics of the disk to MS-DOS has been destroyed. In this case, you may continue CHKDSK processing.
Processing cannot continue (CHKDSK) (FDISK)	There is not enough memory in your machine to process CHKDSK for this disk. You must obtain more memory to run CHKDSK.

▶

ERROR MESSAGE	MEANING
Program too big to fit in memory (MS-DOS)(LINK)	You must acquire more memory to run your application. It is possible that some programs you have run are still using some memory. Try to restart MS-DOS, and retype the command. If you still receive the error message and have used the BUFFERS = *parameter* Directive in the systems CONFIG.SYS file, reduce the number of buffers, re-boot the system and retry the command. However, if you still receive this message, you must acquire more memory.
Read fault (device error)	MS-DOS cannot read the requested data from the named device.
Read error in: filename (Commands)	The command could not read the entire file.
Requested drive is not available (FORMAT)	Self-explanatory.
Requested stack size exceeds 64K (LINK)	You have tried to specify a stack size greater than 64 Kbytes.
Sector not found (device error)	The sector containing the data cannot be found, usually due to a defective area on the disk.
Sector size too large in file *filename* (Startup)	The device sector size defined in the device driver *filename* exceeds the system limit.

ERROR MESSAGE	MEANING
Seek (device error)	The disk drive cannot find the proper track on the disk.
Segment size exceeds 64K (LINK)	You have tried to combine identically named segments resulting in a segment requirement of more than the addressing limit of 64 Kbytes.
Share already installed (SHARE)	Share can only be installed once.
Source and target drives are the same (BACKUP) (RESTORE)	Specify a different source and target drive in your BACKUP or RESTORE command.
Source diskette bad or unusable (DISKCOPY)	You will get this error message if there is a read error from your source diskette.
Specified MS-DOS search directory bad (MS-DOS)	The SHELL directive in the CONFIG.SYS file is incorrect. The place that you have told MS-DOS to find COMMAND.COM does not exist, or COMMAND.COM is not in that place.
Specified drive does not exist, or is nonremovable (DISKCOMP DISKCOPY)	If you referred to your hard disk drive as a parameter of these commands, you will get this error message. Otherwise you are referring to a non-existent drive.
Stack size exceeds 65535 bytes (LINK)	The specified stack size exceeds the system limit.

ERROR MESSAGE	MEANING
Start1, exiting	This message is issued in extreme circumstances, when the system actually halts itself. If this error is returned, you have to reboot the system.
Symbol defined more than once (LINK)	Two or more modules have defined the same symbol name.
Symbol table capacity exceeded (LINK)	Very many, and/or very long names were entered exceeding the limit of approximately 25 Kbytes.
Syntax error (MS-DOS)	You have entered a command in the incorrect format.
System halt	This message is issued in extreme circumstances, when the system actually halts itself. If this error is returned, you have to reboot the system.
SYS cannot install MS-DOS on this disk (SYS)	You cannot use SYS as the target disk does not have an empty root directory or is not formatted with the **/S** option.
Target diskette may be unusuable (DISKCOPY)	After an unrecoverable read, write or verify error, the copy may be corrupted.
Target does not contain backup files (BACKUP)	BACKUP was called with **/A** parameter, but the files on the backup disk are not to be found on the target drive.

ERROR MESSAGE	MEANING
Target is full (RESTORE)	RESTORE cannot continue for this reason.
Terminate batch job (Y/N)? (MS-DOS)	If you press **CTRL-BREAK** or **CTRL-C** while in batch mode, MS-DOS asks you whether or not you wish to end batch processing. Press **Y** to end processing. Press **N** to continue the batch job.
The last file was not restored (RESTORE)	If there is not enough room on the target disk to receive a particular backed up file, RESTORE will delete the partially backed up file. Delete unwanted files on the target disk, then enter RESTORE filename, where filename is the name of the file previously partially restored. Interruption to RESTORE can be continued in the same fashion.
Too many external symbols in one module (LINK)	There are too many external symbols in one module. The limit is 256 external symbols per module.
Too many groups (LINK)	Too many groups are defined. The limit is 10.
Too many libraries specified (LINK)	Too many libraries have been named. The limit is eight libraries.
Too many open files Too many files open (MS-DOS)(EDLIN)	Try to solve this problem by increasing the FILES directive in CONFIG.SYS.

ERROR MESSAGE	MEANING
Too many overlays (LINK)	The system limit of the overlays has been exceeded
Too many public symbols (LINK)	There are too many public symbols in one module. The limit is 1024 public symbols.
Too many segments or classes (LINK)	You have too many segments or classes in your source files. The limit is 256 (segments and classes taken together).
Track 0 bad-disk unusable (FORMAT)	You cannot use the disk you are trying to format because track 0 is damaged.
Tree past this point not processed (CHKDSK)	Track 0 of the disk being checked is damaged. CHKDSK cannot continue.
Unable to create directory (MKDIR)	The specified directory cannot be created due to one of the following conditions: • Directory already exists. • One of the directory path-names could not be found. • The root directory is full.
Unexpected end-of-file on library (LINK)	There is probably an error in the library file

ERROR MESSAGE	MEANING
Unexpected end-of-file on VM.TMP (LINK)	There is probably an error in the library file. The diskette containing VM.TMP is not present in the drive.
Unrecognised command in CONFIG.SYS	You have entered an invalid command in the configuration file.
Unrecoverable error in directory Convert directory to file (Y/N)? (CHKDSK)	If you press **Y** in response to this prompt, CHKDSK will convert the bad directory into a file. You can then fix the directory yourself or delete it. If you press **N**, you may not be able to write to or read from the bad directory.
Unrecoverable file sharing error (SHARE)	Caused by a file sharing conflict.
Unrecoverable read error (DISKCOPY)	Self-explanatory. The diskette has probably been damaged.
Unresolved externals: *list* (LINK)	The external symbols listed were not defined in the modules or library files. Do not attempt to run the file created by the linker.
VM.TMP is an illegal filename and has been ignored (LINK)	You cannot use VM.TMP as an object module.
Warning: directory full (RECOVER)	No more files can be recovered in the directory.

ERROR MESSAGE	MEANING
Warning: diskette is out of sequence Replace diskette or continue. Strike any key when ready. (RESTORE)	Self-explanatory.
Warning: The file above is marked readonly. Replace the file (Y/N)? (RESTORE)	When RESTORE **/P** is specified and the file encountered is read-only; answer **Y** if you want to replace the file or **N** if not. When you type **ENTER**, RESTORE will continue.
Warning: The file above was changed after it was backed up. Replace the file (Y/N)? (RESTORE)	When RESTORE **/P** is specified and the file on the target disk has a later time and date than the same named file on the source disk; answer **Y** if you want to replace the file on the target disk or **N** if not. When you type **ENTER**, RESTORE will continue.
Warning: no files were found to backup (BACKUP)	Check your backup file specifications for incorrect input.
Warning: No files were found to restore (RESTORE)	Self-explanatory.
Warning: no stack segment (LINK)	None of the object modules specified contain a statement allocating stack space, but you entered the **/STACK** switch.
Warning: read error on EXE file (EXE2BIN)	The input file has not been read correctly.

ERROR MESSAGE	MEANING
Warning: Segment of absolute or unknown type (LINK)	There is a bad object module or an attempt has been made to link modules that the linker cannot handle; for example, an absolute object module.
Write error in TMP file (LINK)	No more disk space remains to expand the VM.TMP file.
Write error on RUN file (LINK)	There is not enough disk space for the Run file.
Write fault (device error)	MS-DOS cannot successfully write data from/to the name device.
Write protect (device error)	You have tried to write data to a disk that is write-protected.
Write Protected (DISKCOPY)	You cannot use a write protected diskette for a copy.
XXXX error on file *yyyy* (PRINT)	A disk error of type *XXXX* occurred when reading data from file *yyyy*. This message is listed on the printer and printing then stops.

ERROR MESSAGE	MEANING
XXX lost clusters found in *YYY* chains Convert lost chains to files (Y/N)? (CHKDSK)	If you respond **Y** to this prompt and had specified the **/F** switch, CHKDSK will recover the lost blocks it found when checking the disk. CHKDSK will create a directory entry and a file for you with the filename FILEnnnn.CHK. If you respond **N** and had specified **/F** switch, CHKDSK frees the lost blocks so they can be reallocated.

E. GLOSSARY OF TERMS

ABOUT THIS APPENDIX

This appendix describes some technical terms, whose understanding is necessary for the user of MS-DOS.

CONTENTS

GLOSSARY OF TERMS E-1

The following table defines the terminology in this manual.

TERM	MEANING
active partition	The partition on hard disk which contain the operating system files enabling the bootstrapping of the computer. This happens on system reset or when the computer is turned on.
ASCII	American Standard Code for Information Interchange. A 7 BIT code, which has been extended to an 8 BIT code (a BYTE) to represent graphic characters and international characters.
basic input output system (BIOS)	Part of the operating system which provides an interface with the machine hardware. Most of the BIOS is in Read Only Memory (ROM), the rest is loaded from the system disk.
binary digit (BIT)	In a binary numbering system, only two marks are used 0 and 1. Each of these marks is called a binary digit.
bootable file	A file of a specific format that the bootstrap loader can load into memory to initialize the system.
byte	Eight bits, which is normally a code for an ASCII character.
current directory	The directory in which you are working.
cylinder	Hard disks usually consist of a number of platters. A cylinder refers to the same track on each surface of the platters which form a notional cylinder.

TERM	MEANING
disk	A diskette or hard disk.
diskette	A single or double-sided 5 1/4in. floppy disk.
drive specifier	A letter referring to the diskette drive or hard disk drive in question. For example it may be: A - first diskette drive. B - second diskette drive. C - hard disk drive.
editing function keys	The keys that invoke the intra-line commands.
external command	A command that is not loaded into memory at initialization. Such commands reside on disk from where they are loaded, executed and purged from memory.
formatting	Disks must be formatted before they can be used with MS-DOS. Formatting places tracks, which are split into sectors, onto the surface(s) of a disk. The sectors are of fixed number of bytes, typically 512 bytes. Also formatting places a boot record and an empty directory on the disk.
hard disk	A disk that is built into the computer. A hard disk can store much more information than a floppy disk, and the computer can retrieve information from it faster.

TERM	MEANING
hardware reset	A system reinitialization caused by pressing the physical reset button. The subsequent initialization includes diagnostic tests and a reset of all system parameters. Any AUTOEXEC.BAT file or CONFIG.SYS file is executed.
inter-line commands	The EDLIN commands that operate on entire lines of text.
intra-line commands	The commands invoked by the special editing function keys that perform editing operations within a single line of text.
internal command	A command that is embedded in the COMMAND.COM file and resides in memory whenever MS-DOS is booted.
Kilobyte KB	2 to the power 10 = 1024 Bytes
Mega-byte MB	2 to the power 20 = 1 048 576 Bytes
nil parameter	A parameter to a command where the parameter in question is not specified in the command line. The parameter therefore assumes a default value.
partition	A certain number of cylinders of a hard disk, which have been set aside for the use of a particular operating system. That operating system treats the partition like a complete, but smaller, hard disk. The maximum partition size allowed for MS-DOS is 32 MB. The number of cylinders this corresponds to, depends on how many bytes per cylinder.

▶

TERM	MEANING
pathname	A sequence of one or more directory names separated by backslashes optionally beginning with a drive specifier and optionally terminating in a file name. It specifies a path through a directory structure to access a file or directory.
sectors	The track on a disk is divided into sectors. MS-DOS disks are soft sectored. The number of sectors per track is typically 8, 9 or 15.
source line	A line of text containing either the last command line entered or the current line in a file being edited. It can be retrieved in whole or part or modified using the special editing function keys.
system file	An MS-DOS file that is present on the MS-DOS system diskette that contains system software. There are three such files: - MSDOS.SYS (hidden file) - IO.SYS (hidden file) - COMMAND.COM
system reset	A system reinitialisation caused by pressing the **CTRL**, **ALT** and **DEL** keys simultaneously. Any AUTOEXEC.BAT file or CONFIG.SYS file is executed.
text file	An ASCII file whose records are separated by CR/LF.

▶

TERM	MEANING
tracks per inch (t.p.i.)	A disk track is the circular locus of the head as the disk rotates. The head can be moved to the other tracks; they are concentric circles. A double density diskette has 48t.p.i. A quad density disk has 96 t.p.i.
virtual disk	An emulation of backing store in Random Access Memory (RAM). It is faster than disk backing store, but the information on virtual disk is lost when the computer is turned off.
volume label	A name that can be assigned to a disk by the FORMAT command. It will subsequently be displayed in a directory listing, or by the VOL command
wild card character	A special symbol used to represent any single character (?), or any string of characters (*).
working session	The time between booting MS-DOS and the next boot of MS-DOS or switch-off.

F. THE VIRTUAL DISK SYSTEM

ABOUT THIS APPENDIX

This appendix describes how to install VDISK.SYS.

CONTENTS

INSTALLING VDISK.SYS F-1

A virtual disk is part of Random Access Memory which emulates a backing store Disk. The VDISK.SYS is a device driver, which when installed, enables a virtual disk drive with the next available drive letter. For example if you have a two physical drive machine, with drives A: and B:, when VDISK.SYS is installed, you will have an extra drive C:. The only difference between virtual disk and real disk is that when you turn your machine off, the information on virtual disk will be lost. So remember to COPY all files you want to keep from virtual disk to a real disk, before you turn your machine off.

INSTALLING VDISK.SYS

VDISK.SYS is a file included on your MS-DOS Diskettes. To install the VDISK console drive the following command must be placed in the CONFIG.SYS file.

DEVICE = [*d:*][*path*] **VDISK.SYS** [*disk-size*][*sector-size*][*entries*] [**/E**[: *max*]]

Where

SYNTAX ELEMENT	MEANING
d	Specifies the drive where VDISK.SYS is to be found.
path	Specifies the directory where VDISK.SYS is to be found.
disk-size	A decimal value declaring the virtual disk size in Kilo Bytes. The value be from 1 through to the maximum free Random Access Memory on your computer. The default is 64 Kilo Bytes.

SYNTAX ELEMENT	MEANING
sector-size	A decimal value declaring the sector size in bytes. The value may be 128, 256 or 512. The default is 128 bytes.
entries	A decimal number declaring the maximum number of directory entries required for files. One entry is used for a volume label. Three entries are used for each sub-directory. The value may be from 2 through 512. The default is 64 entries.
/E	This switch specifies that the VDISK driver uses ''extended memory''. MS-DOS can only directly address 640 KB. Random Access Memory installed over 640 KB is ''extended memory''. This option is only available on Intel 80286 based Personal Computers, do not use this switch for other Personal Computers.
:max	This is a parameter of the **/E** switch, specifying the maximum number of sectors (of sector-size) that are transferred to/from extended memory. The possible values are a decimal number in the range 1 through 8. The default value is 8.

For example place the following assignment in CONFIG.SYS.

```
DEVICE= C:\BIN\VDISK.SYS 128 512 32
```

Rebootstrap your computer and a message similar to the following will be displayed.

```
VDISK Version 3.01 virtual disk D:
      Directory entries adjusted
      Buffer size: 128 KB
      Sector size: 512
      Directory entries: 32
```

Refer to Appendix C for more details of CONFIG.SYS.

Remarks

The following situations cause VDISK to fail to install.

- less than 64 KB free memory.
- using the **/E** switch with no or insufficient extended memory.

In these cases the following message is output on the screen.

```
VDISK not installed insufficient memory
```

VDISK.SYS might adjust the parameters you specified in the following ways:

PARAMETER	ADJUSTMENT
entries	Rounded up: 32 bytes (per file entry) multiplied by entries (rounded up) equals a multiple of sector-size. This is so as not to waste space available for directory file entries. Rounded down: one sector at a time times sector-size divided by 32 bytes, until there are sufficient sectors to hold the File Allocation Table, the directory file entries and at least two sectors for files. When the number of sectors for directory file entries has been rounded down to one, the above error message is issued. In this case redeclare the VDISK parameters in CONFIG.SYS and rebootstrap the system.

▶

PARAMETER	ADJUSTMENT
disk-size	Rounded down: so as to leave 64 KB Random Access Memory free. If this is not possible the above error message is issued. In this case you need to buy more Random Access Memory in order to use VDISK.

Remarks

You can install more than one virtual disk by placing several **DEVICE = VDISK.SYS** commands in your CONFIG.SYS. Each virtual disk takes the next available drive letter. If the next available drive letter is F, for use by virtual disk, place the command **LASTDRIVE = F** before your DEVICE command in CONFIG.SYS. Refer to Appendix C for more details.

INDEX

A

ANSI escape sequences, B-1
ANSI.SYS device
(CONFIG, SYS), B-1, C-5
ASCII tables, A-1
ASSIGN command, 6-8
ATTRIB command, 6-6, 6-10
AUTOEXEC.BAT, 1-2, 2-9, 5-14
AUTOEXEC.BAT file, 2-10

B

back-up diskettes, 2-5
BACKUP command, 6-11
batch processing, 1-2, 5-10
batch processing commands, 5-11
boot-strapping MS-DOS, 2-1
booting up MS-DOS, 2-1
BREAK command, 6-14
BREAK command
(CONFIG.SYS), C-1
BUFFERS command
(CONFIG. SYS), C-2

C

changing directory, 4-14
CHDIR (CD) command, 4-12, 4-14, 6-15
checking disks and files, 4-15
CHKDSK command, 4-15, 6-17
CLS command, 6-20
COMMAND command, 6-21
command formats, 5-1
commands
  ASSIGN, 6-8
  ATTRIB, 6-6, 6-10
  BACKUP, 6-11
  BREAK, 6-14
  CHDIR (CD), 4-12, 4-14, 6-15
  CHKDSK, 4-15, 6-17
  CLS, 6-20
  COMMAND, 6-21
  COMP, 6-24
  COPY, 4-5, 6-6, 6-27
  CTTY, 6-34
  DATE, 6-35
  DEBUG, 10-1
  DEL, 6-38
  DIR, 4-9, 6-6, 6-40
  DISKCOMP, 6-41
  DISKCOPY, 6-45
  ECHO, 6-47
  EDIT (video file editor), 7-1
  EDLIN (line editor), 8-2
  ERASE, 6-47
  EXE2BIN, 5-7, 6-48
  EXIT, 6-50
  external, 5-6, 5-8
  FC, 6-51
  FDISK, 2-13
  FIND, 5-18, 6-61
  FOR, 6-64
  FORMAT, 2-16, 6-67
  GOTO, 6-70
  GRAFTABL, 6-75
  GRAPHICS, 6-72
  GWBASIC, 6-76
  HEXDUMP, 6-77
  IF, 6-80
  internal, 5-6
  JOIN, 6-82
  LABEL, 6-87
  LINK, 5-7, 9-1
  MKDIR (MD), 4-14, 6-88
  MODE, 6-89
  MORE 5-18, 5-19, 6-96
  PATH, 4-11, 6-97
  PAUSE, 5-11, 6-99
  PRINT, 6-100
  PROMPT, 6-104
  RECOVER, 6-107
  REM, 5-11, 6-109
  REN, 6-109
  RESTORE, 6-111

RMDIR (RD), 4-15, 6-115
SELECT, 6-116
SET, 6-118
SHARE, 6-4, 6-120
SHIFT, 6-122
SORT, 5-18, 6-123
SUBST, 6-125
SYS, 6-129
TIME, 6-130
TREE, 6-132
TYPE, 4-6, 6-133
VER, 6-134
VERIFY, 6-135
VOL, 6-136
WRITECHK, 2-4, 6-137
COMP command, 6-24
CONFIG. SYS
definition, C-1
CONFIG.SYS commands
BREAK, C-1
BUFFERS, C-2
COUNTRY, 2-11, C-3
DEVICE, C-5
FCBS, C-5
FILES, C-6
LASTDRIVE, C-7
SHELL, C-8
configuring MS-DOS, 2-11
control keys, 3-1
Control-Break, 5-10, 6-14, C-1
Control-C, 5-10, 6-14, C-1
Control-Number Lock, 5-10
Control-S, 5-10
COPY command, 4-5, 6-6, 6-27
COUNTRY command (CONFIG.SYS), 2-11, C-3
creating a new directory, 4-14
CTTY command, 6-21, 6-34
cursor movement (ANSI.SYS), B-2

D

DATE command, 6-35
DEBUG command, 10-1
default drive, 2-2
definition of terms, E-1
DEL command, 6-38
DEVICE command (CONFIG.SYS), C-5
device errors, D-1
device names, 4-4
DIR command, 2-8, 4-9, 6-6, 6-40
directories
changing, 4-14
creating, 4-14
definition, 4-6
displaying, 4-12
hierarchical, 1-1, 4-7
multilevel, 1-1
removing, 4-15
root, 4-8
sub-directories, 4-7
disk operating system, 1-1
DISKCOMP command, 6-41
DISKCOPY command, 2-6, 6-45
diskettes
back-up, 2-5
formatting, 2-3
handling, 1-4
labelling, 1-4
purchasing, 1-4
write-protection, 1-4
displaying the current directory, 4-12
drive
capacities, 1-3, 2-4
default, 2-2
identification, 1-3
single, 2-3
drives
High-Capacity, 2-4
Normal-Capacity, 2-4

E

ECHO command, 6-47
EDIT command
(video file editor), 7-1
editing function keys, 3-4
EDLIN (line editor)
command, 8-2
ENTER Key, 1-7, 3-1, 5-10, 7-7,
8-5
ERASE command, 6-47
error messages (MS-DOS), D-1
examining files, 4-6
EXE2BIN command, 5-7, 6-48
EXIT command, 6-50
external command syntax, 5-8
external commands, 1-2, 4-11, 5-6

F

FC command, 6-51
FCBS command (CONFIG.SYS),
C-5
FDISK command, 2-13
filenames definition, 4-1
files
  examining, 4-6
  naming, 4-1
  protecting, 4-6
  validity, 4-15
FILES command (CONFIG.SYS),
C-6
FIND command, 5-18, 6-61
fixed disk setup program, 2-12
font tables, A-1
FOR command, 6-64
FORMAT command, 6-67
FORMAT command (hard disk),
2-16
formatting diskettes, 2-3

G

GOTO command, 6-70
GRAFTABL command, 6-75
GRAPHICS command, 6-72
GWBASIC command, 6-76

H

HEXDUMP command, 6-77
hidden files, 2-9
hierarchical directories, 4-7

I

IF command, 6-80
input, 5-16
input redirection, 5-17
internal commands, 1-2, 5-6

J

JOIN command, 6-82

K

key reassignment
(ANSI.SYS), B-7
Keyboard Code Tables, A-1

L

LABEL command, 6-87
LASTDRIVE command
(CONFIG.SYS), C-7
line editor, 8-1
LINK command, 5-7, 9-1

M

MKDIR (MD) command, 4-14, 6-88
MODE command, 6-89
MORE command, 5-18, 5-19, 6-96
MS-DOS commands, 5-9

N

national characters, A-1
national keyboard drivers, 2-9
national screen fonts, 2-9
networking, 6-5
notation convention, 1-5

O

output, 5-16
output redirection, 5-16

P

parameters
  definition, 5-2
  nil, 5-5
PATH command, 4-11, 6-97
pathnames
  definition, 4-9
  maximum length, 4-11
  syntax, 4-10
PAUSE command, 5-11, 6-99
piping, 5-18
PRINT command, 6-100
programming
  FORTRAN, 1-1
  GWBASIC, 1-1
  Macro Assembler, 1-1
  PASCAL, 1-1
PROMPT command, 6-104
protecting files, 4-6

R

real-time clock, 1-2
RECOVER command, 6-107
redirecting input, 5-17
redirecting output, 5-16
REM command, 5-11, 6-109
removing a directory, 4-15
REN command, 6-109
replaceable parameters, 5-15
reserved device names, 4-4
resident external commands, 1-2
RESTORE command, 6-111
RMDIR (RD) command, 4-15, 6-115
root directory, 4-8

S

screen handling (ANSI.SYS), B-1
SELECT command, 6-116
SET command, 6-118
SHARE command, 6-5, 6-120
SHELL command (CONFIG.SYS), C-8
SHIFT command, 6-122
single drive systems, 2-3
SORT command, 5-18, 6-123
source drive, 5-10
sub-directories, 4-7
SUBST command, 6-125
SYS command, 6-129

T

target drive, 5-10
TIME command, 6-130
TREE command, 6-132
TYPE command, 4-6, 6-133

V

VER command, 6-134
VERIFY command, 6-135
video file editor, 7-1
VDISK.SYS device
(CONFIG.SYS), C-5, F-1
virtual disk, F-1
VOL command, 6-136
volume label, 2-4

W

wild cards, 4-2
  the * wild card, 4-3
  the ? wild card, 4-2
WRITECHK command, 2-4, 6-137

NOTICE

Code 4024230 R (0)
Printed in Italy

OLIVETTI
PERSONAL
COMPUTER

olivetti